PACIFIC PACKET RADIO SOCIETY NEWSLETTER

volume 2 number 2 February 1986

AJ6T>PPRS:

FCC BOMBSHELL HITS PACKET COMMUNITY!

The FCC has released a Report and Order on PR Docket 85-105 which liberalizes Part 97 rules dealing with automatic operation on digital modes above 50 Mhz. The bad news is that the FCC refused to change other rules which prohibit the automatic retransmission of third party traffic. If this ruling is upheld, severe restraints may be imposed on bulletin boards and digipeaters. Our budding nationwide packet switching network may be crippled from the start. Why is the prohibition of automatic retransmission of third party traffic a problem? When a packet message is sent from station A to station B via a bulletin board (in a store and forward fashion) or via a digipeater (in near real time) there are three stations involved in the transmission of the message (station A, station B, and the PBBS/digipeater). From the point of view of the PBBS or digipeater, it is handling third party traffic, since it is transmitting on behalf of someone other than its control operator. Here is the catch: the PBBS or digipeater is not allowed to transmit third party traffic automatically, hence a CONTROL OPERATOR MUST BE PRESENT to supervise the transmissions! Imagine how impractical it is to have a control operator continuously present to monitor the operation of every bulletin board and digipeater. Digipeaters and PBBS would become sporadic, in operation only when their control operators were home or at least monitoring via a radio or telephone link. The dream of a full-time nationwide packet switching network could never be realized with this limitation.

A detailed understanding of the FCC regulations regarding control operator requirements and third party traffic requires careful study of exisiting regulations and the new Report and Order. Some parts of the regs are vague and self-contradictory. There is room for interpretation. A debate is now raging among amateur operators as to what the FCC said, and what the implications are for packet operation.

Existing FCC rules require that every amateur radio station must have a control operator who is responsible for the operation of that station. Except when the station is operating automatically, this operator must be present at the control point (either physically at the transmitter or running the station via a remote control link). Prior to this new Report and Order, automatic operation was only permitted for beacon stations (operating in the beacon subbands), repeater stations (operating in the repeater subbands), or stations in auxillary operation (for example, remote bases). The new rule change allows automatic operation of all stations using a digital mode (packet, RTTY, AMTOR, etc.) above 50 Mhz (subject to the third party restrictions). This makes it legal (finally) for the various "auto answer" routines which have become very popular ("I am not here right now, please leave a message"). It is now also legal to send automatic beacons (but still not a good operating practice) or to program your computer to automatically send files directly to a friend late at night while you are asleep. The FCC requires that "devices and procedures" must be implemented during automatic operation to insure that all other regulations are still complied with (for example, proper ID, in-band transmissions, low spurious emissions, no malicious interference, good operating practice, etc.).

Third party traffic is defined in Part 97.3 (v) as "amateur radio communication by or under the supervision of the control operator at an amateur radio station to another amateur radio station on behalf of anyone other than the control operator." This is a very broad definition. The FCC seems to contradict itself in paragraph 3 of the Report and Order where they state: "...repeater stations are already permitted to be operated under automatic

control. However, as with any amateur station, when they are transmitting third-party traffic, the control operator must be present at the control point monitoring and supervising the transmissions." Yet according to the definition of third party traffic, it seems that ALL repeater transmissions involve third party traffic. If that is true, then no repeater would be allowed to operate automatically. Are all voice repeaters illegal too? A clarification is obviously needed. NTS and ARES traffic is clearly third party in nature, and seems to be banned from all automatic unattended repeaters (both voice and digital types).

What is the FCC trying to accomplish with this limitation on third party traffic? Apparantly they want to prevent amateur frequencies from being flooded with messages from non amateur third parties. They also "do not want to give our approbation to a mechanism which could be used to circumvent the international Radio Regulations which forbid exchange of amateur third-party traffic between countries who are not parties to agreements permitting such traffic." These are valid goals, but they are separate issues not directly connected with amateur-to-amateur traffic inside this country.

What action should be taken? Should all PBBS stop operation immediately? Should all digipeater owners tear their machines from the hilltops? Should we wait and do nothing? Keep in mind that as far as the third party rules are concerned, NOTHING HAS CHANGED! The FCC is not making new rules on this point, they are merely affirming old rules. VHF Mailboxes are no more illegal today than they were last month or last year (HF mailboxes are probably in trouble though since the FCC did not permit automatic operation at all below 50 MHz, and the ARRL is opposed to HF automatic operation because of congestion). I doubt that the FCC is going to start issuing Notices of Violation in the immediate future. Lets continue to operate just as we have while we petition the FCC to change the regulations. The PPRS will file a Petition for Reconsideration with the FCC. Come to the next PPRS meeting on Tuesday, February 4, 1986 to discuss these issues and help formulate a reply to the FCC. The direction of future growth of packet radio will be determined in the next few months. I hope we can convince the FCC not to kill this mode in its infancy.

...AND IN OTHER NEWS:

Jim Tittsler, AI8A, will speak at the next PPRS meeting. He will describe the common commands for several popular TNCs and explain what they mean.

WESTNET continues to grow. New paths have opened up into Oregon (via W6AMT,W6AMT-7,WA6YNG-1,W6SDP,WA7KSW-1,WB7SZM-1) and Utah (!) (via WESTNET, AA6TN-1,WA7HXO-3,WA7GTU-1).

The TAPR annual meeting will be held in Tuscon on February 8, 1986. Get your ballots for the TAPR Board of Director's election in before that date.

Please pay your PPRS dues for 1986! Most of the dues is spent on this newsletter, and we cant keep going without your support. Your mailing label indicates whether you are a current member (86 indicates paid up for this year, 85 means this is your last newsletter).

73, Walter E Miller, AJ6T 28Jan86

M 5142 Tom Clark (W3IWI,2976) 1/24/86 2:19 AM L:247
KEYS:/FCC REPORT AND ORDER 85-105/AUTOMATIC CONTROL DOCKET/FULL TEXT/
TO: (Group 95)

Art Feller, KB4ZJ obtained the complete Report and Order on FCC 85-105, the new rules concerning automatic control at frequencies above 30 MHz. None of us have had time to read it in detail, so the following is forwarded without interpretation. Art offered his apologies for any typos, since he had to bang it in by hand. I say thanks to Art from all of us for making the material available in such a timely manner. 73, Tom

==========

))))))) FORWARDED FROM THE W3IWI PBBS (((((

Via W3IWI From KB4ZJ: Msg# 1761, Rcvd 860124/0537z, Sent 860124/0552z

Before the
Federal Communications Commission
Washington, DC 20554

PR
FCC 86-18
36413

In the matter of)
Amendment of Part 97 of the) PR Docket No. 85-105
Commission's Rules to permit)
automatic control of amateur) RM-4879
radio stations.)

REPORT AND ORDER

Adopted: January 13, 1986; Released: January 16, 1986

By the Commission:

1. On April 5, 1985, the Commission adopted a Notice of Proposed Rule Making (50 F.R. 15196; April 17, 1985) to permit any amateur station to be under automatic control provided that operation was on frequencies above 29.5 MHz and that no third-party traffic was transmitted. This was an expansion of the proposal requested by The American Radio Relay League, Inc. (ARRL), who had requested automatic control only for stations transmitting digital communications while operating on frequencies above 30 MHz. Nineteen comments and one reply comment were filed in this proceeding. [1]

2. Our Notice of Proposed Rule Making in this proceeding, supra, particularly invited amateur radio operators experienced in automatic control to submit comments on the practicality of expanding automatic control to encompass all amateur operations, not just digital communications. Such comments were not forthcoming. Because of the lack of user support, we will adopt only the ARRL's recommendation to limit automatic control to digital communications on very high frequencies (VHF) and above. (We select 50 MHz rather than 30 MHz, as the petitioner requests, because there are no amateur frequencies between 29.7 and 50 MHz.)

---

[1] Comments files by the American Radio Relay League, Inc., (ARRL), Robert C. Clements, Jess de la Cuesta and Joseph Anthony Wolos were filed late. Consideration of the viewpoints expressed in those comments will aid in the resolution of this proceeding. Therefore, we accept them.

==========

3. In the Notice, supra, we reminded amateur operators that the current rules require the presence of the control operator at the station control point whenever third-party traffic is being transmitted. We emphasized that the proposed amendments would still prohibit automatic control of the station when it is transmitting third-party traffic. Some commentors were confused about unsupervised third-party traffic. For example, repeater stations are already permitted to be operated under automatic control. However, as with any amateur station, when they are transmitting third-party traffic, the control operator must be present at the control point monitoring and supervising the transmissions.

4. Many of the commentors request that high-speed digital operating modes, such as packet-switching, bulletin boards, computer based message systems and electronic mail boxes be exempt from the requirement that the control operator supervise third-party traffic. They believe that the third-party rules as applied to such high-speed digital communications are impractical and would, in effect, nullify the advantages of automatic control. To acquiesce in that request would be inconsistent with other types of amateur operation. Third-party traffic is radio communications on behalf of anyone other than the control operator. [2] Neither the speed at which the message is transmitted nor the emission (voice, telegraphy, digital etc.) changes its character. This was pointed out in our letter of October 19, 1978, to Richard Baldwin, then General Manager of the ARRL. [3] In that letter, we reiterated that unsupervised third-party traffic by amateur stations is not permitted.

5. Some comments suggest that the third-party traffic rules be amended so that they would be applicable only at the time the third-party traffic is introduced into the amateur communications system. However, screening the message content at its introduction does not change the character of the traffic. It is still third party traffic which must be closely regulated in the non-common carrier Amateur service. Otherwise, amateur facilities and frequencies would be open to non-amateurs and could eclipse other amateur uses. Moreover, we are also concerned about the final destination of the message. We do not want to give our approbation to a mechanism which could be used to circumvent the international Radio Regulations which forbid exchange of amateur third-party traffic between countries who are not parties to agreements permitting such traffic.

---

[2] Section 97.3(v).

[3] FCC 78-742; 70 F.C.C. 2d 1303. See also News Release No. 2028, October 25, 1978.

International third-party radio communications are prohibited by Section 97.114 of the amateur rules and Article 32, number 2733 of the International Radio Regulations (Geneva, 1979), except where arrangements have been made between the two countries involved, Article 32, number 2734.

6. Some commentors suggested that MF and HF frequencies between 1.8 and 29.5 MHz be added to the frequencies available for automatic control or that automatic control be extended at least to all digital communications below 29.5 MHz on a regular basis or by temporary special authority (STA). They state that coast-to-coast coverage for point-to-point message handling would be accommodated by including MF and HF frequencies. Because of the possibility of congestion on the MF and HF frequencies, we do not believe that it would be advisable to permit automatic control on those frequencies.

7. Robert C. Clements is under the impression that we inserted a clarification into proposed Section 97.79(b) that the station licensee is presumed to be the control operator of the station, unless there is documentation to the contrary. However, this is essentially the same as the present wording of Section 97.97(b). The words "at all times" will be deleted from this rule in order to be consistent with the revised wording of Section 97.3(m)(3).

8. For the reasons given herein, we amend our rules to permit automatic control only for digital communications on amateur frequencies 50 MHz and above. Further, we affirm our present rule that requires the control operator to be present at the control point whenever the station is engaging in third-party traffic.

9. IT IS ORDERED, That Part 97 is amended as set forth in the Appendix hereto. This action is taken persuant to the authority contained in Sections 4(i) and 303(r) of the Communications Act of 1934, as amended.

10. IT IS FURTHER ORDERED, That these rule amendments shall become effective, March 14, 1986.

11. IT IS FURTHER ORDERED, That the Secretary shall cause a copy of this Report and Order to be published in the Federal Register.

12. IT IS FURTHER ORDERED, That this proceeding is terminated.

13. Information in this matter may be obtained by contacting Maurice J. DePont, (202) 632-4964, Private Radio Bureau, Federal Communications Commission, Washington, D.C. 20554.

FEDERAL COMMUNICATIONS COMMISSION

William J. Tricarico
Secretary

Attachment: Appendix

Part 97 of Chapter I of Title 47 of the Code of Federal Regulations is amended, as follows:

1. The authority citation for Part 97 continues to read as follows:
   Authority citation: 48 Stat. 1066, 1082, as amended;
   47 U.S.C. 154, 303.

2. Section 97.3(m)(3) is amended to read:
   Section 97.3 Definitions.
   * * * * * *
   (m) * * *
   (1) * * *
   (2) * * *
   (3) Automatic control means the use of devices and procedures for control without the control operator being present at the control point when the station is transmitting.

3. Section 97.79(b) is amended to read:
   Section 97.97 Control operator requirements.
   * * * * * *

(b) Every amateur radio station, when transmitting, must have a control operator. The control operator must be present at the control point of the station, except when the station is transmitting under automatic control. The control operator must be a licensed amateur radio operator or permittee designated by the station licensee. The control operator and the station licensee are both responsible for the proper operation of the station. For purposes of enforcement of the rules, of this part, the FCC will presume that the station licensee is the control operator of the station, unless documentation to the contrary exists.

==========

4. Section 97.69 is amended by adding a new paragraph (d), as follows:
Section 97.69 Digital communications.
* * * * * *
(d) An amateur station may be under automatic control when transmitting digital communications on frequencies 50 MHz and above.

5. A new section 97.80 is added, as follows:
Section 97.80 Operation under automatic control.
(a) When under automatic control, devices must be installed and procedures must be implemented which will ensure compliance with the rules when the control operator is not present at the control point of the amateur station.
(b) No amateur station may be operated under automatic control while transmitting third-party traffic.
(c) Automatic control of an amateur station must cease upon notification by the Engineer-in-Charge of a Commission field office that the station is transmitting improperly or causing harmfull interference to other stations. Automatic operation must not be resumed without prior approval of the Engineer-in-Charge.

6. Section 97.114 is amended by addiding a new subparagraph (4) to paragraph (b) as follows:
Section 97.114 Third-party traffic.
* * * * * *
(b) * * *
(4) Third-party traffic from an amateur station under automatic control.

* * * * * *

-- END --

Coordinators Column

George, wb6ral
January 15, 1986

News of Note: Growth of 145.03!

There are several 145.03 digipeaters for Users to use for linking. n6dbt-2 is the only coordinated one... though several others are "in the mill".

Coverage on 145.03 for Northern California is quite good as many of you have discovered. There are a couple of high level digipeaters (wb6lpz-1 in the Santa Cruz mountains, and w6bxn (moved from .01) on Mt. Bullion over Turlock way).

Additionally, there is a large and growing network of lower level machines placed so as to reach into all "pockets" of activity. (Pockets of packets?).

There are also a couple of mailboxes that "visit" .03 and announce themselves. These are not all the "regular" kind and it is interesting to use a mailbox that is not just a "wOrli clone".

Another Good Feature of 145.03 (as it is now) is that it is a good "local ragchew" channel:

> Unlike .01, which by plan has high level machines designed to link the state... and thus fills with QSOs Real Easily.
>
> And, also unlike 145.09MHz, (the channel with 800 mailboxes), which is rapidly reaching the point of saturation (if it isn't there already).

145.03MHz is a good local LAN in the best sence of the term:

> Supported with digipeaters that have few "holes" in their coverage.
>
> Presently not crowded with mailbox traffic, or,
>
> lots of long distance QSOs.

This time last year there was NO ONE on 145.03. Now, it is a fully supported and active channel. Packet traffic on the two meter channels is growing. "New" channels are filling up everyday; where are we going to be a year from now?

Enjoy!

Next step... 220 Backbone!

This column has refered obliquely to something called the "220 backbone". What is it? What can it do for you? Read on...

What IS It?

The "220 backbone" is merely shorthand for a bunch of dual ported digipeaters being planned. One of the ports will be on two meters and look like those digipeaters/mailboxes that you have grown to know and love. The Other port is on a common 220MHz frequency and operates to link these dual-port machines to each other (sort of a network just for the machines). This 220MHz port will (hopefully) operate at 9600 baud. This is eight times as fast as the 1200 baud links everyone is using now and even faster than that in dynamic operation.

One figure is worth 1024 words:

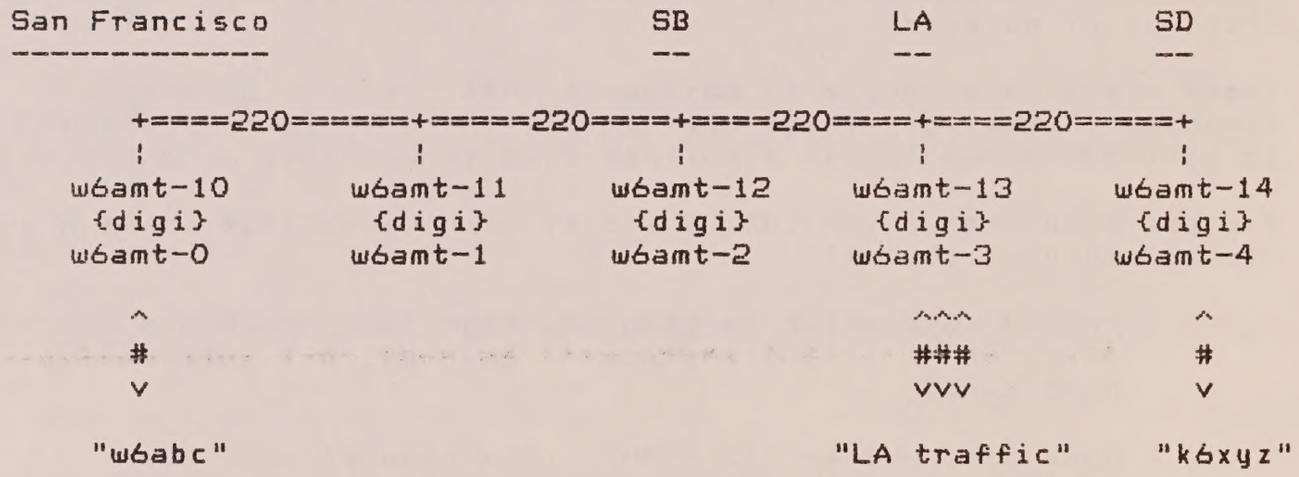


Figure 1. Dual Port Topology

What It Will Do For You

Since there will now be another, faster "network" capable of carrying data and which can tie into two meter digipeaters without regard to the two meter frequencies, there are several advantages that will be realized:

1) Network Robustness will improve because there will be more possible ways to "get somewhere".

2) Network speed will improve because the 220MHz half will run at 9600 baud.

3) Users will be able to tie into the network on several frequencies; you will be able to pick the two meter path with the least traffic - a step in solving the congestion now on two meter User input frequencies.

4) High traffic two meter LANs can be avoided (now, if you want to talk to San Diego, you HAVE to go THROUGH the Los Angeles traffic (usually without success)).

How To Use It

Using the "new" dual port digipeaters involves some new tricks which have been decided upon to provide an elegant solution to the "only eight hops" problem. The thing to remember is that you address a dual port digipeater be the call you want TO COME OUT OF IT FROM. See "figure 1" above and follow the bouncing ball...

to get from w6abc (you) in SF to k6xyz in SD:

"c k6xyz v w6amt-10 w6amt-11 w6amt-12 w6amt-13 w6amt-4"

note that the two meter INPUT port is NOT specified in the path but that the two meter OUTPUT port is. Interesting. Note also that there are now path permutations too numerous to mention here so that you don't have to give up at the first "retry out".

How Best To Use It

Some of the techniques for best utilizing this "new" capability when it comes on-line have been mentioned above. Certainly "going around" possible two meter packet bottlenecks is an obvious technique. Other uses will require that you really know "Westnet". There will be possibilities to take advantage of long haul hops on one two meter frequency and then use the dual port guy(s) to "change frequency" to get onto the mailbox of your choice.

Also, realize that there will be dual port (or, multiple port) digipeaters on other bands (like six meters, or, HF) that might get you a lot farther with more reliability. The skillful packeteer will be able to "play" the Network for a lot more data bandwidth than now exists.

Who Has Committed?

Another way of saying this is, "When and Where are we going to see these things?". The answer: Real Soon Now. Those familiar with these types of Amateur (read "spare time") efforts know not to expect anything on schedule. There ARE people working on this technology all over the country. People have these things WORKING. From "figure 1" above, it looks like the AMT people are planning to put up some "dual porters". There are enough others wanting to put their mailbox or digipeater on that it looks like there will be a significant amount of hardware up and going by mid year. Remember, this time last year...

What YOUR PPRS Is Doing For You (and Packet Radio).

Since you are reading this, I assume that you are at least partially familiar with the PPRS Newsletter. There are Other Groups that have cleverer names for their "rag", but, the PPRS newsletter is by far the best of the bunch.... (talk about unprovable!)

Bringing You DRnet is another real nice feature from PPRS. DRnet is a nationwide forum for Packet Radio that seems to collect some very interesting discussions and ideas. It is even interesting when printed so small...

Providing a forum for diverse Packeteer community is another function for the PPRS. Members from all over Northern California provide a truly representative group. There is seldom unanamity, but almost always a consensus.

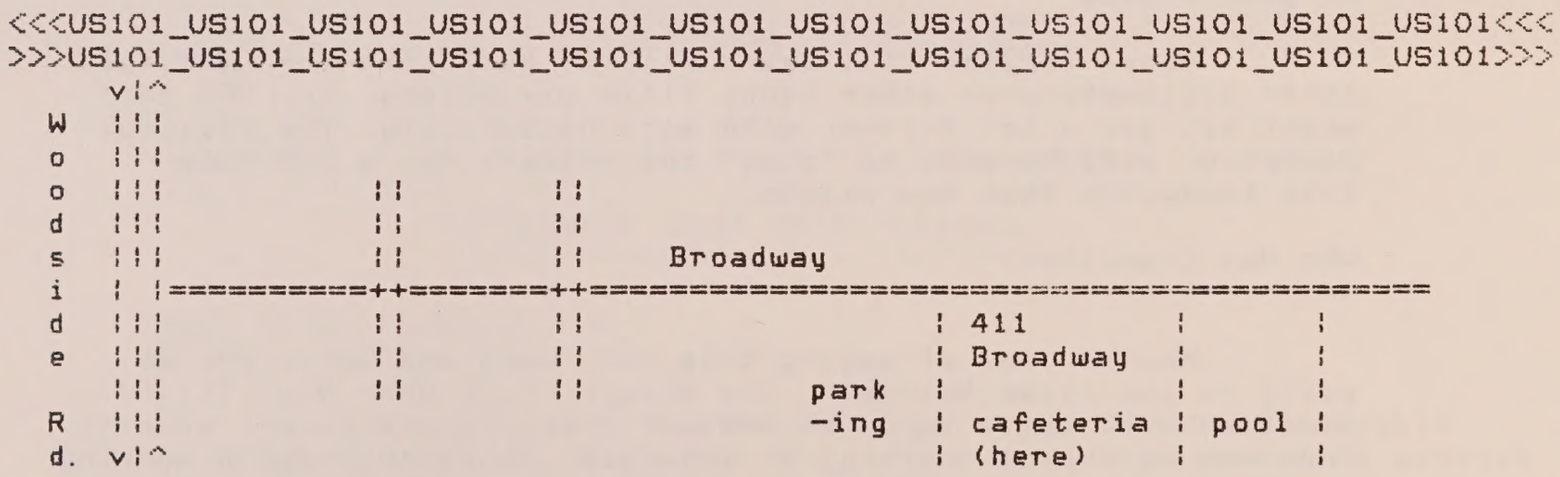
Oh, I almost forgot, ANOTHER thing that your PPRS tries to do (just like a real ham club) is having club meetings...

MAP TO PPRS MEETINGS

Here is a map to the:

"FIRST TUESDAY OF THE MONTH AT 7:30PM"

PPRS meeting.



Take U.S. 101 (Bayshore Freeway) to Woodside Road (84), in Redwood City. Turn Left on Broadway (real close to 101).

Go past two stops to 411 Broadway (on the right).

It is the Cafeteria located next to the reflecting pool on Ampex Corporate property.

END OF (cute) MAP TO PPRS MEETINGS

All for just $18.00/yr!

Next Month: Execution servers, Gripes, an updated Westnet map, and the first complete list of Coordinated digipeaters - Ever!

a software package to perform these functions. Currently, we have psuedo-coded 30 software modules for TEXNET. Certain rules and programming metholodogy can help to ease the construction difficulties with a complex problem.

---

[1] I forgot to credit the fundamental work of the subject in one of my prior articles.
"Computer Networks" by Andrew S. Tannenbaum; Prentice-Hall, Inc. C 1981.

C2974 CC107 Phil R. Karn (ka9q,2979) 12/ 6/85 7:27 PM L:61
KEYS:/TEXNET COMMENTS/

A few comments on Part 5 of the TEXNET articles that was just posted.

It is interesting to watch as various members of the amateur community slowly re-invent each aspect of the ARPA Internet protocols, step by step, even if they do not realize that this is what they are doing. With a few differences I'll talk about later, this is what TEXNET seems to be doing.

The use of encapsulation to send a network datagram across a series of links is fundamental to how the Internet operates. TEXNET seems to be doing the same thing, but it assumes that the only link level protocol that will ever be used is AX.25. Why? There are lots of link level protocols out there, and if you can encapsulate your datagrams in all of them transparently, you've created a much more flexible way of doing things. The link protocol can be customized for the particular situation (i.e., providing hop-by-hop acks if they are deemed necessary), while the higher level network protocol can worry about things like global addressing and routing. I believe that the two need to be kept completely separate, with the network level datagram service assuming as little as possible about the underlying link level protocol.

The article points out the fact that user interfacing with the network is clumsy, because you need some way to tell the network what to do with your packets. This is very true! Eventually it should become obvious that the way to solve this problem is to simplify the user/network interface as much as possible, i.e., by the provision of a simple datagram interface. This also means that the end-to-end protocol or "transport" protocol, for which we now use AX.25 LAPB even though we continue to call it a "link" protocol will have to reside in the user's TNC. I believe that the only practical way to do this is by adding the full-blown network transport protocol (i.e., TCP) to each user's TNC. I have shown with the code I've recently written that this can be done in about 6K bytes, which is probably smaller than just the command processor that would be required by a connection-oriented network interface.

The later comments about the difference between congestion control and flow control are very true. I've been trying to get this point across for some time. One thing that should be apparent from the discussion is that "backpressure" flow control schemes are to be avoided whenever possible, since they allow a user to leave packets unreceived in the network for indefinite periods of time. If a user isn't ready to receive a packet, it shouldn't be sent in the first place. All end-to-end transport protocols, such as TCP and AX.25 as used presently, have flow control windows to provide this. Link level flow control, if used at all, should be used only to provide hop-by-hop control over traffic, not as a substitute for end-to-end flow control. A well-designed transport protocol will also attempt to generate as few packets as possible to carry a given amount of data, and this is something that modern versions of TCP (with the "Nagle Rule") do very well. They also react to special messages from the network that indicate congestion by backing off at the rate at which they inject traffic into the network.

In summary, then, I would argue strongly that the TEXNET design team take a close look at the Internet protocols, since their overall design aims and approaches appear to be compatible with those of TEXNET. I would be happy to provide any documentation if required.

73, Phil

C2974 CC108 Howard Goldstein (N2WX,2987) 12/ 8/85 2:30 PM L:75
KEYS:/I THINK I CAN I KNOW I CAN I THINK I CAN/
A: 107

Finally there is just so much BULLSHIT and SLANDER that its time to correct some of these statements!

Its a crock of shit that "6K bytes is probably smaller than just the command processor that would be required by a connection oriented network interface." I don't know what kind of programming techniques prevail in the TCP/IP world but if it takes 6K bytes, someone needs to go back to school. Is anyone else tired of pontifications from the ivory tower - in the real world, not this high flake intelluctal crap that we've got reams of paper wasted over now, things are not as clearcut as you've alledged them to be - damn they aren't even LIKE the way you've alledged them. A VC implementation takes around 6K bytes for everything after ax2512v2 - X.25 l3, X.75, the command interface, the whole kit and kaboodle.

But are we here to argue about coding efficiency? Nooooo, not if memory is so cheap

No matter how you cut up the datagramme approach we're gonna wind up with a HIGH FLAKE NETWORK OF DIGIPEATERS.

[In case no one's haven't noticed, there is already a 'standard' link level protocol. Sure there are lots of others - so how in hell did the STUPID and INCONSISTENT "UI" handling manage to get in to pervert this otherwise pristine standard? Has anyone ever tried to reconcile the imbecilic way UI frames are supposed to be handled in L2V2 with the rest of the document? Kind of makes one wonder how standards get formulated. This case is a perfect example of the "klugeyness" inherent in the datagramme approach. What a mess, a new piece added whenever someone gets a new idea they like. No two people running the same thing. More later

]

Transports- Very nice, lets put a transport protocol in each switch and then we won't need a link protocol. Nifty, but it's still digis, plus everyone has to have some complex and redundant transport protocol in their TNC. Can we see some code running in a TNC? AX.25 LEVEL 3 EXISTS - TODAY - AND RUNS IN A TNC OVER AX.25 LEVEL 2.

Everyone needs to remember that the TCP/IP bunch wants us to use a connectionless network. Can you all imagine trying to CONNECT to someone VIA 20 or 30 digipeaters? That's the approach being touted, friends, its end-to-end all the way - WHETHER YOUR SESSION REQUIRES END-END ACKS OR NOT you're gonna have to use them.

I think this should be a LOCAL choice a user could specify if he wants it.

Yes, congestion is quite different from flow control, and if a user isn't ready to receive a packet the question is not leaving packets unreceived in the network (again, memory is supposed to be cheap - right???) but how long it takes for a sender to realize and get data through again after he's turned back on. Takes a while for the "clogged" end to turn the other side back on with TCP/IP, no? Yes? Flow control is an exception ax.25 l3 plans for. It's designed to make buffer management VERY EASY no matter how many VCs go across an interface (okay, mod256 might be desirable for bunches of them) and optimizes throughput for everyone concerned.

In fact, I personally found it much easier to install AX.25 level 3 and its practically identical IP cohort X.75 than level 2 was! Someone working with a high level language should be able breeze through the Network layer in a week. Terry's proposal was definitely ahead its time. I think its time has now arrived.

On congestion control: How does a stateless switch deal with priorities and congestion? Correct me if I'm wrong, I'm under the impression that it TRASHES stuff it can't deal with without any indication. Just like what we're doing now with digis. May we have an example of how a stateless switch acts when its plate is full?

In summary, there seems to be a fundamental irreconcilable difference between the datagramme and VC approaches to network. I contend this difference has to do with the expectations each community has of it's media.

Datagrammites expect an infinite # of links to be error prone and rarely available (perhaps as a result of nuclear detonations), in a way similar to the way one would try to have a QSO in UNPROTO mode.

A virtual circuit network expects a finite number of error free links to be available. Links like that which are already provided by AX.25 L2V2.

This difference in expectations has direct effect on the capabilities of the network.

Planning in a datagramme world is like doing 5mph under speed limit on a coast-coast drive the whole way. No one does it, despite the better chances of living through the trip and not getting a ticket.

And those folks with the $$ fly their own learjets exactly where they have to go without concerning themselves with drunk drivers and local yokels.

I do not expect the amateur community to be able to afford learjets (or high speed leased lines between switches), nor do I think there's a use for a network convinced every transmission was puked on, corrupted, and otherwise violated.

73 Howie

C2987 CC244 Howard Goldstein (N2WX,2987) 12/ 1/85 3:59 PM L:10
KEYS:/8K RELEASE/
A: 243

Tom, I really would rather not have a seperate 8K release just to keep things easy to maintain.

However there is probably a need to support the unmodified boards around so there will be one more release to kind of tie up the loose ends where there isn't a need/desire to upgrade to rev2.

If they can't wait a little while longer go ahead and let them have the release. But they're on their own since there's known faults in in that last 8K.....

73 Howie

C2987 CC245 Howard Goldstein (N2WX,2987) 12/ 3/85 10:34 AM L:73
(ORIG.) 12/ 3/85 1:23 AM L:16
KEYS:/1.1.2 AVAILABLE/EXTENSIVE SOFTWARE RELEASE NOTES/+READ THIS ITEM FOR .HEX/

To: Beta
Fm: Howie N2WX
Re: 1.1.2 notes
Dist: Beta

Please +READ C2987 CC245 for the .HEX of the release described herein.

1.1.2: Changes in this release are virtually all attributable to feedback from Beta testers. Therefore, if you like anything in it, pat yourselves on the back. This is as much your release as it is anyone's!

Time to drop some names. Especially thank the following individuals (in no particular order) for their input is represented by the features and fixes of this release: KV7B & D, WA7GXD, W3IWI, WB6YMH, AD7I, and KY2D. (Is your call missing from this list? How about downloading a release and making some comments?!)

Here are notes releated to 1.1.2:

FIXES occupy much of these notes. The following has been repaired, I think:

o- Flow release causing data loss of xmit data

o- Anamolous disconnect when in S16 link state

o- Retry-out handling when version 2 of 12 works like table now

o- ID doubling fixed (i think -- could not dupe the bug here)

o- Sluggish carrier detect repaired, the TNC is much friendlier to other stns now

o- Monitor headers aren't displayed for received data from a link when MCON ON and MON ON

o- NULLS are handled in int routines, all combinations supported w/o loss

o- CTEXT packetized at connect time instead of traveling through the queues

o- NULL artifact from true-break transparent escape is now captured

o- rare corruption of data with "cmd:" repaired

o- 10 second timers (BEACON, CHECK) work at 10 sec intervals now.

o- and finally, a very troubling intermittent that caused innumerable problems like more loss of link state, strange stuff in cmd parameters etc.

CHANGES not result of bugs:

o- ^Y handled much faster (inside async int routine)

o- UA responses while in link setup mode are IGNORED if they differ from the attempted path

o- Number-only calls allowed to support links to AX.121NA switches...

o- The date of assembly (mm/dd/yy) appended to the release number

ex: Release 1.1.2 12/03/85 - 16K RAM

o- Interrupt handling speeded up at least two times. Support for 9600 both sides, fulldup is gauranteed with standard (2.45Mhz clock). Support for 19.2Kbaud also assured with appropriate hardware mods for 4.9Mhz and faster I/O clocks.

(My tests used two TNC2's, both running 9600 sync and async, FULLDUP ON MAX 7 PACL 0 in trans and CONV mode (ECHO ON and OFF) could not drop characters thru a link of six fulldup digipeaters and a 28K file.)

o- The overflow buffer expanded to 256 characters (overflow after XOFF sent)

NEW FEATURES:

o- Failure of a CONPERMed link results in a

"*** LINK OUT OF ORDER, possible data loss [opt. daytime stamp]" message

o- the command LCStream ON|OFF [default ON] added.

When LCStream is ON, the character immediately following the streamswitch character is converted to upper case before being acted upon. When off, the case is significant.

o- the command NOmode ON|OFF [defualt OFF] added.

When the parameter is ON, the TNC will -NEVER- jump on its own between cmd: and conv or trans, or vice versa. Only user commands (CONV, TRANS, or ^C) may change the typein mode.

If the NOMODE parm is OFF then mode jumping is handled according to NEWMODE and the rev 1 TNC 2 manual.

That's it for 1.1.2. As usual, all comments are appreciated, and most receive immediate attention.

Thank you everyone for your invaluable, visible (and I hope, continuing) input to this project. 73 Howie

PS: releases prior to 1.1.2 are NOT gauranteed for 9600 baud, and should not be relied upon in any mode at this data rate!

M 21226 GARY GARRIOTT (VITA,568) 12/ 9/85 10:18 AM L:52
KEYS:/ON TO ETHIOPIA!/
TO: N2DSY, W2VY, (Group 95)

THOMAS AND GORDON--THIS IS THE STOCK STATEMENT THAT HAS GONE OUT TO OTHERS THAT PREVIOUSLY EXPRESSED INTEREST....SEND ME YOUR QTH IF STILL INTERESTED SO I CAN SEND YOU PROJECT DESCRIPTION...THANX, GARY

9 December 1985

Dear OM:

Some months ago you responded to an inquiry regarding a possible demonstration project in Ethiopia using packet radio. I am pleased to announce that preliminary approval has been acquired from the Ethiopian government to conduct the project.

Attached is a brief project description. Three TAPR TNC2's are being acquired along with Radio Shack Model 200 computers to be used in the demonstration sites. These units will be "retrofitted" on existing Motorola HF SSB transceivers.

Given the nature of the project it is very important to have tested the system in advance of actually arrriving in Ethiopia. This is because the system should be viewed as fully functional with as little experimentation as necessary while actually in the country (government officials will no doubt be looking over the consultants' shoulders the entire time). this probably means about a week's worth of testing here in the us.

Some preliminary training of both American and Ethiopian personnel will also be conducted during the three weeks or so that the two consultants will be in the country.

Please be advised that this is a volunteer activity, although all expenses (airfare, meals, lodging) will be paid through VITA. We would like to do the project as soon after the holiday season as possible.

If you are still interested, please contact me as soon as possible. If you have a recent resume or curriculum vitae, forward that as well.

Needless to say, the use of packet radio technology in disaster/relief communications is exciting and could solve a host of problems associated with existing HF radio. To my knowledge, this will be the first demo of packet anywhere in the Third World and will generate a huge amount of interest if successful.

73,

Gary Garriott (WA9FMQ)
Manager
Information Technology

M 21392 Rich Amundson (WA0JFS,2965) 12/ 9/85 10:47 PM L:25
KEYS:/SAREX2 TESTS/
TO: (Group 95)

TO: GROUP 95(PARTICULARLY MIDWEST)

FROM: RICH AMUNDSON

RE: SAREX2 TESTING

The flight test of the SAREX2 hardware hopefully taking place Wednesday December 11 will be using the call of N0BKB-1 (listen for beacons for last minute changes) and will have at least 25 watts of rf power available to partially make for the lack of a 100 mile altitude. We will be about 10,000 feet msl around the Greenfield, IA airport which has a location of Latitude 41-19-36N and a Longitude of 94-26-52W. We will be going in shallow circles

to try and prevent shadowing of the antenna by the aircraft. W0RPK will be conducting a coordination net for anyone with questions on the AMSET 80 Meter frequency during the flight and will be in contact with the aircraft.

We invite all who can hear the beacons to try and connect with us to see if the system can be (ab)used. Our downlink frequency will be 145.55 and uplink _600 or the standard repeater split of 144.95.

Hope to hear many of you on packet Wednesday from about 1930 CST until
.

I HOPE THIS CAME OUT AL RIGHT. Telenet did a job on it. 73's
Rich Amundson WA0JFS

M 21432 Phil R. Karn (ka9q,2979) 12/10/85 2:34 AM L:10
KEYS:/NONTRANSPARENT MAILERS/
TO: (Group 95)

To all DRNET users:

Please avoid generating lines that consist of a single period at the beginning of a line. Several other electronic mail systems (e.g., Telenet Telemail and ARPA Internet mail) treat such lines as "end of message" with annoying results when you're trying to relay messages.

Thanks,
Phil

M 22030 GARY GARRIOTT (VITA,568) 12/11/85 2:25 PM L:6
KEYS:/VOLS FOR TNC 2 CONSTRUCTION/
TO: (Group 95)

WE ARE LOOKING FOR VOLUNTEERS WHO MIGHT BE INTERESTED IN PUTTING TOGETHER THREE (3) TAPR TNC 2 KITS FOR THE ETHIOPIA PACKET DEMO PROJECT. ANY TAKERS? CAN YOU THINK OF A BETTER WAY TO SPEND YOUR CHRISTMAS BREAK??

JUST LEAVE A MSG IF INTERESTED....THANX AND 73, GARY

M 22065 Pete Eaton (WB9FLW,2970) 12/11/85 3:48 PM L:8
KEYS:/HAZELTURKEY FIXED..BACK ONLINE/
TO: (Group 95)

Hi Gang!

Just a note to let you know I got the Hazelturkey terminal working again, and am online once again.

Did I miss any good fight!?

Pete

M 22219 Rich Amundson (WA0JFS,2965) 12/12/85 12:22 AM L:18
TO: (Group 95)

TO: GROUP 95
FROM: RICH AMUNDSON
RE: SAREX2TESTING

WELL, THE FLIGHT WENT FAIRLY WELL HERE TONIGHT IN SPITE OF SOME GLITCHES WITH THE AIRPLANE. WE HAD TO CUT THE FLIGHT SHORT BUT IN ONE HOUR OF OPERATION WE HAD 515 CONNECTS AND WE DID SEE ON THE ONBOARD M100 THAT UP TO 8 CHANNELS WERE ACTIVE SIMULTANEOUSLY. MORE DETAILED INFO WILL BE AVAILABLE WHEN W0RPK HAS A CHANCE TO PERUSE THE DISK FILE CLOSER. IT DOES ACCUMULATE DATA AS HE TOLD US HE HAD 80K WITHIN THE FIRST HALF HOUR.
HOWIE, WE FOUND NO PROBLEMS AT ALL WITH THE SOFTWARE FROM THE OPERATORS POINT OF VIEW AND MORE WILL BE TOLD ON EXAMINATION OF THE FILES.
TOM, LARRY, N0BKB, HAD A REPORTER OUT AT THE AIRPORT BEFORE WE LEFT (FROM THE LOCAL NEWSPAPER) WHO TOOK SOME PICTURES SO WE WILL FORWARD SOME AND THE NEWS COPY AS SOON AS IT IS AVAILABLE.
73,S AND FILM AT 10 RICH WA0JFS

C2973 CC48 Thomas A. Moulton (W2VY,995) 12/10/85 11:34 AM L:21
KEYS:/A DIFFERENT VC BETWEEN EACH NODE IN A CONNECTION FROM USER TO USER/
A: 47

Jack,
I think you misunderstand where X.25 VC's start and end.

In a X.25 based network (commerical or not) an X.25 link is made between two nodes (stations) and between those stations a VC (or PVC if you must) can be created. [so far so good right?]

That case was from A to B, now let's look at the normal case, more stations.

Now A wants to connect to C, A looks in it's routing tables and sees that C can be gotten to from B (I'll spare us all the X.121 addresses) so A does a Call to B with C's address, this sets up a VC from A to B, then B calls the next station (D say) and sets up a VC TOTALLY INDIPENDENT OF WHAT IS GOING ON WITH A.

The limit of 4096 is NOT the number of VC's in the Entire network but the limit of the number of calls BETWEEN two stations (ie A to B)

As far as the routing tables required for X.121 addresses, we would have to have an entry for each area code and also entries for our local nodes The number will be much less than the 4000 entries you talked about.

C2973 CC49 Phil R. Karn (ka9q,2979) 12/10/85 1:08 PM L:13
KEYS:/AMATEUR VC THROUGHPUT/

I have a question for those building X.25-based networks. What will the frame size and packet window limits be? I am concerned about the throughput that will be available through a single virtual circuit.

I am resigned to the development of VC-based networks, so I'm thinking now about how they'll be integrated as subnets of the Internet-based network I'm building. Based on my experience at work with Telenet, considerable gyrations are required (like opening multiple, parallel virtual circuits to the same destination and distributing datagrams among them) because of the tiny packet and window size limits. I hope the amateur VC networks, if built, do not repeat the same mistake.

Phil

C2973 CC50 Phil R. Karn (ka9q,2979) 12/10/85 2:41 PM L:52
KEYS:/IS LAPB REALLY NECESSARY?/

It is generally agreed that we need hop-by-hop acknowledgments in amateur packet radio in order to get reasonable performance. It has occurred to me, however, that this does NOT necessarily imply that there needs to be a "connection" between each pair of adjacent nodes.

Current thinking has been to use the full features of AX.25 level 2 to provide a connection-oriented link between each pair of nodes. This path could be used either to support a higher level virtual circuit, or to reliably pass datagrams from one packet switch to another. The amount of complexity in the full AX.25 level 2 has long bothered me. Recently I completed an implementation of the ARPA TCP transport level protocol (the one considered by many to be "too complex" for amateur use) and discovered that it was in fact an EASIER job than my earlier implementation of AX.25 Level 2. I think something's wrong here.

Fortunately, there are "escape clauses" in AX.25 that allow a higher layer to bypass the connection-oriented part (LAPB), namely the UI frame. Ordinarily the UI frame is sent as a pure datagram, i.e., there is no acknowledgment sent by or expected from the receiver. However, the last revision of AX.25 Level 2 specified that should the poll (P) bit be set in a UI frame, a UA response is expected from the other end. It seems to me that this feature provides a way to get link level acks without having to initiate connections and maintain all of the associated state.

Since a packet switch based on AX.25 Level 2 is likely to have many simultaneous connections because of the rich connectivity at the local level, maintaining a large number of protocol control blocks (many of which will be idle most of the time) seems a wasteful use of memory. It seems a better idea when passing IP datagrams from one switch to another to simply send them in UI frames with the P bit set in order to request a link level ack from the other end. The link layer on the sending end must then only hold onto the frame long enough to receive an acknowledgement, or to retransmit the frame until one is received. No memory need be tied up unless there is actual activity on the link, and the LAPB portion of AX.25 need not be implemented.

The only problem with this approach is that there needs to be as many acknowledgment packets as there are data packets. However, recent techniques in congestion control suggest that packet windows be reduced to 1 outstanding packet at a time anyway in order to prevent "congestion collapse" and to avoid clobbering a returning acknowledgement with another data packet. This also avoids the ambiguity caused by the lack of sequence numbers or other indication in the returning UA frame as to which packet is being acknowledged. As long as the packets themselves are allowed to be a reasonable size, good throughput can still be obtained.

Comments or suggestions? I'm going to persue this idea as I continue my TCP/IP-on-AX.25 implementation, and I'd appreciate constructive help.

73, Phil

C2973 CC51 Phil R. Karn (ka9q,2979) 12/11/85 12:19 AM L:186
KEYS:/TCP PROGRAMMERS MANUAL/PLEASE DISTRIBUTE FOR COMMENT/

My work on a TCP implementation for amateur packet radio has reached the point where I can describe the interface provided to the application by TCP. I have also written a UDP (User Datagram Protocol); however, this is not at the same level of maturity as TCP and is therefore more subject to change. This note is meant primarily as "advance information" to any implementers considering writing applications.

To review the purpose of TCP: it supports a reliable, sequenced, byte stream "connection" on an end-to-end basis. It fits in roughly at the Transport layer (level 4) of the OSI model. Since a single TCP module supports multiple connections through the use of port numbers, it also provides Session layer (level 5) functionality without the need for a distinct protocol. (Or it makes the session layer unnecessary, depending on your point of view). This package is written as a "module" intended to be compiled and linked with the application(s) so that they can be run as one program on the same machine. This greatly simplifies the user/TCP interface, since it becomes just a set of internal subroutine calls on a single machine. Reliability is much greater, since a hardware failure that kills TCP will likely take any applications with it anyway. Only IP datagrams flow out of the machine across hardware interfaces (such as asynch RS-232 ports or whatever else is available) so hardware flow control or complicated host/front-end protocols are unnecessary.

A TCP connection is uniquely specified by the concatenation of source and destination "sockets". In turn, a socket is the concatenation of a host address (a 32-bit integer) and a TCP port (a 16-bit integer), defined by the C structure

```
struct socket {
long address;/* 32-bit IP address */
short port;/* 16-bit TCP port */
};
```

Therefore it is possible to have several distinct connections established at the same time to a single port on a given machine, as long as the source sockets are distinct. Port numbers are used either through mutual agreement, or more commonly when a "standard" service is involved, a "well known port" number. For example, to obtain standard remote login service (known as "telnet") one initiates a connection to TCP port 23; to send mail using the Simple Mail Transfer Protocol (SMTP) one talks to port 25. ARPA maintains port number lists and periodically publishes them. They will also assign port numbers to a new application on request if it appears to be of general interest.

TCP connections are best modeled as a pair of one-way paths (one in each direction) rather than as a single full-duplex path. Station A may close its path to station B leaving the reverse path from B to A unaffected. B may continue to send data to A indefinitely until it too closes its half of the connection. This is known as "graceful close" and can greatly simplify an application.

My TCP code supports five basic operations on a connection: open, send, receive, close and delete. A sixth, tcp_state(), is provided mainly for debugging. They are summarized in the following section in the form of C declarations and descriptions of each argument.

```
int net_error;
```

This global variable is used to indicate the specific cause of an error in one of the TCP or UDP functions. All functions returning integers (i.e., all except open_tcp) return -1 in the event of an error, and net_error should be examined to determine the cause. The possible errors are defined as constants in a header file.

```
/* Open a TCP connection */
struct tcb *
open_tcp(lsocket,fsocket,active,notify,tos)
struct socket *lsocket,*fsocket;
int active;
void (*notify)();
char tos;
```

"lsocket" and "fsocket" are pointers to the local and foreign sockets, respectively.

"active" is Ø for a "passive" open (one in the TCP LISTEN state). A passive open does not cause any packets to be sent, but enables TCP to accept a subsequent active open from another TCP. If a specific foreign socket is passed to a passive open, then connect requests from all other foreign sockets will be rejected. If the foreign socket fields are set to zero, then connect requests from any foreign socket will be accepted. If "active" is 1, TCP will initiate a connection to a remote socket that must previously have been created in the LISTEN state. The foreign socket must be completely specified in an active open.

"notify" is an optional receive "upcall" mechanism, useful when running in a non operating system environment. If "notify" is non-zero, it is taken as the address of a function to be called whenever a "significant" amount of data arrives. This user-provided function may then invoke recv_tcp() to obtain the incoming data.

"tos" is the Internet "type of service" field, consisting of precedence and class of service parameters. There are 8 levels of precedence, with the bottom 6 defined by the military as Routine, Priority, Immediate, Flash, Flash Override and CRITICAL. (Two more are available for internal network functions). For amateur use we can use the lower four as Routine, Welfare, Priority and Emergency. Three more bits specify class of service, indicating that especially high reliability, high throughput or low delay is needed for this connection. The entire TOS field is passed along to IP in each datagram and is interpreted by each IP gateway (packet switch) in the route. The precedence value actually used is the higher of those specified in the two tcp_open() calls.

open_tcp() returns a pointer to an internal Transmission Control Block ("tcb"). This "magic cookie" must be passed back as the first argument to all other TCP calls. In event of error, the NULL pointer (Ø) is returned.

The only limit on the number of TCBs that may exist at any time (i.e., the number of simultaneous connections) is the amount of free memory on the machine. Each TCB on a 16-bit processor takes up about 129 bytes; additional memory is consumed and freed dynamically as needed to buffer send and receive data. Deleting a TCB (see the delete_tcb() call) reclaims its space.

```
/* Send data on a TCP connection */
int
send_tcp(tcb,data,cnt)
struct tcb *tcb;
char *data;
unsigned cnt;
```

"tcb" is the pointer returned by the open_tcp() call. "data" points to the user's buffer, and "cnt" specifies how long it is. The number of bytes actually queued for transmission is returned; it will equal "cnt" unless some sort of error occurs, such as lack of memory for buffering. The data is copied into an internal queue, a transmission is attempted, and the function returns so that the user may immediately reuse his buffer.

TCP uses positive acknowledgments and retransmission to ensure in-order delivery, but this is largely invisible to the user. TCP enforces no limit on how much data can be queued for transmission, so the user should be careful not to run the system out of free memory. (This is something else that requires a multitasking kernel to do right).

```
/* Receive data on a TCP connection */
int
recv_tcp(tcb,data,cnt)
struct tcb *tcb;
char *data;
unsigned cnt;
```

The arguments to recv_tcp() are identical to those of send_tcp(), except that any data on the connection's receive queue is placed in the user's buffer, up to a maximum of "cnt" bytes. The actual number of bytes received (the lesser of "cnt" and the number pending on the receive queue) is returned. Since this TCP module cannot assume the presence of sleep/wakeup primitives provided by an underlying operating system, recv_tcp() is currently designed to return -1 with net_error set to EWOULDBLK if no incoming data is pending. The "notify" feature on open_tcp() is provided to eliminate the need for constant polling of the recv_tcp() function; whenever TCP calls the notify function, it guarantees that recv_tcp() will not return -1. (Technical note: "notify" is called whenever a PUSH or FIN bit is seen in an incoming segment, or if the receive window fills. It is also called before an ACK is sent back to the remote TCP, in order to give the user an opportunity to piggyback any data in response.)

When the remote TCP closes its half of the connection and all prior incoming data has been read by the local user, subsequent calls to recv_tcp() return Ø rather than -1 as an "end of transmission" indicator.

```
/* Close a TCP connection */
close_tcp(tcb)
struct tcb *tcb;
```

This tells TCP that the local user has no more data to send. However, the remote TCP may continue to send data indefinitely to the local user, until the remote user also does a close_tcp(). An attempt to send data after a close_tcp() is an error.

```
/* Delete a TCP connection */
delete_tcp(tcb)
struct tcb *tcb;
```

When the connection has been closed in both connections and all incoming data has been read, this call is made to cause TCP to reclaim the space taken up by the TCP control block. Any unread incoming data is lost.

```
/* Dump a TCP connection state */
tcp_state(tcb)
struct tcb *tcb;
```

This debugging call prints an ASCII-formatted dump of the TCP connection state on the terminal. You need a copy of the TCP specification (ARPA RFC 793 or MIL-STD-1778) to interpret most of the numbers.

Well, that's it. Constructive comments on this interface are welcomed.

73, Phil Karn, KA9Q

C2973 CC52 Phil R. Karn (ka9q,2979) 12/11/85 10:32 PM L:60
KEYS:/TCP/IP FOR MACINTOSH INFO/

Jack (and others), I've obtained the Internet package for the MacIntosh that I mentioned earlier. It runs a little over half a meg with sources and binaries (in encoded ascii form). I know nothing about this package other than what you read here, which is the original announcement.
Phil

I have posted the sources and binaries for the Macintosh Internet Protocols to net.sources.mac . The sources run about 12,000 lines of code, and another few thousand lines of binhex'd binaries. The source and binaries are stored in twelve shell archive files; note that "sh" and not "csh" should be used to unpack the archives, and that you should do the unpacking

in a new, clean directory. The first archive file absolutely must be unpacked first; after that, the order doesn't matter.

The Internet Protocols are network communication protocols defined by the Department of Defense, in wide use around the world. The Macintosh versions run on Appletalk, the low-cost local area network for the Macintosh computer (and, using a Centram board, for the IBM PC). They are meant not so much for Macintosh to Macintosh communication as for high-speed, error-checked communication with minicomputers and mainframes, such as VAXen and TOPS-20 systems. Some form of Seagate-compatible router will be needed in order to use these programs as they are intended to be used. (Seagate, the Stanford Ethernet Appletalk Gateway, is a family of routers defined by compatible software; the Seagate source code may be retrieved by FTP from SUMEX, in the INFO-MAC archives. Those of you who are not able to FTP should send mail to Bill Croft, su-safe!croft, for information on the tape distribution.)

The programs included are TFTP (simple file transfer), TELNET (remote terminal emulation), CUST (to customize your Macintosh's IP address and so on), and SFMTEST, a test program provided as an illustration of the multiple file "Get File" dialog. Both sources and binaries are provided for all these programs.

The sources may be of interest even to people who have no use for Appletalk, because they contain software for emulating DEC vt100 terminal graphics, error reporting to the user, running multiple tasks simultaneously (actually synchronously) within an application, setting timers at a higher level than direct manipulation of the vertical retrace queue, and an extended "SFGetFile" (standard file package "Get File" dialog) that can be interfaced to from any language without translating the source code, and which allows multiple files to be selected. The sources are in Lisa Pascal, but Lisa Pascal can readily be transliterated into any of the Mac C compilers.

The code originated with the MIT PCIP network package for the IBM PC, written in C. Initial translation was done by Mark Sherman of Dartmouth; I continued this work. The final product is far more than a translation, being thoroughly integrated into the weird and wonderful Macintosh environment, and incorporating a set of, oh, roughly aleph-null bug fixes and major extensions.

Second draft user-level documentation may be found in the file "user" in the first file of the twelve files in the posting. Better documentation is currently being written; by someone else, since I have moved on to other projects, such as LaserWriter print spooling on a UNIX machine. There is currently no external documentation of the source files, though they are reasonably well commented.

-=-

Tim Maroney, Professional Heretic, CMU Center for Art and Technology
tim@k.cs.cmu.edu | uucp: {seismo,decwrl,ucbvax,etc.}!k.cs.cmu.edu!tim
CompuServe: 74176,1360 | God is not dead; he just smells funny.

C2974 CC109 Phil R. Karn (ka9q,2979) 12/ 9/85 2:45 PM L:216
KEYS:/IF YOUVE GOT THE NICKEL/

Howie, I think we have some misunderstandings about what we mean by certain terms when discussing network architectures.

"Command processor" was probably not the correct word to use. What I was referring to is any software module whose function it is to act apon messages from a "client" and possibly respond to them in some way to provide "service". If the "client" is a human sitting at a keyboard typing ASCII strings and parameters, this is what you usually think of as a "command processor". However, the "commands" might be coming from another computer, in which case you might want to design the "command set" (both syntax and semantics) in an entirely different fashion. Of course, a "command processor" that talks to a machine instead of a human might better be called a "protocol module", but then again a module that understands ascii strings typed by humans really IS implementing a "protocol" as well.

What I'm trying to say here is that "command interpreters" and "protocol implementations" are conceptually the same thing: two processing entities, with or without state, that exchange messages according to some rules (i.e., protocols). In this sense, a TNC command set is a full blown protocol. It is a well established fact that protocol designs that involve a large amount of state (memory)

tend to be much more complex, difficult to implement and to debug than those that maintain little or no memory between messages. The problem often becomes overwhelmingly complex when more than two entities are involved and they must all maintain consistent views of each other's state across lossy communication paths. Minimizing the number of places where state must be kept is a practical necessity. Managing virtual circuits across a series of switches, passing call setup and teardown messages back and forth, and handling exception conditions is clearly much more difficult than simply routing packets in stateless switches and confining the notion of a "connection" (if needed) to the endpoints. Just yesterday at the Digital Committee meeting I watched Eric Scace go through a single call setup and teardown operation in Q.931 (the network protocol for ISDN). His flowcharts spanned 16 pages, with 54 (!) messages flying back and forth.
TCP's state diagram fits on a single slide, and it only runs at the endpoints of a network, not in the middle. Three end-to-end messages get you connected, and three end-to-end messages get you disconnected.

I'm sorry you feel that TCP/IP is "high flake intellectual crap" from the ivory towers, because there exists strong evidence to the contrary. An operational Internetwork consisting of dozens of different network types (including Packet Radio) and literally tens of thousands of hosts of all flavors that all manage to interoperate with each other has existed for almost five years. That section of the Internet within my own company consists of several dozen heterogeneous networks at several locations, supporting several hundred hosts used on a day-to-day basis to do real work that we get paid real money for. If that isn't "real world" enough for you, there are implementations of the Internet protocols available for just about every major operating system, and commercial vendors sell these for real bucks.

People buy them because they WORK. Not overhyped, years-away-from-here "vapornets" like ISO and MAP and ISDN, but stuff you can buy off the shelf NOW, plug in, turn on and use. Much of this is the result of TCP/IP being a DoD spec, but many organizations (such as my own) who have nothing to do with the government use it anyway because it works just as well for our needs.

I agree, I don't care much about memory space because memory is cheap (unless you run into a hard limit like 64K address spaces). But program size also tends to be program complexity, and that's certainly something I'd like to minimize. I asked for the TNC-2 programming model because my next step is to port my TCP to the TNC-2. Others are already busy porting it to a variety of machines, and applications people are looking at putting "BBS-like" functions on top of it. Next task is to port some full-blown IP packet switch code I have to the 820 and TAPR NNC.

I fail to understand your contempt for the UI frame. It makes perfect sense when you view AX.25 for what it really is - a two-layer protocol with a VC-oriented pseudo-transport protocol (LAPB) on top of a source-routed pure datagram protocol (the address fields). The UI frame is simply a way to bypass LAPB and get directly at the datagram sub-layer. Nobody forces you to use it, but it's extremely handy when you need a broadcast mechanism -- something that is difficult or impossible to implement efficiently on top of a VC interface.

Regarding transport protocols - your statement is a non-sequitor. Transport and link level protocols do separate jobs, and both are needed even if one of them turns out to be trivial. The real issue is how the various FUNCTIONs need to be allocated among the layers. My view (which is widely shared -- see Salzer's paper "End-to-End Arguments in System Design) is that the end-to-end level (i.e., the transport level) is the ONLY place where you can begin to guarantee "correctness" (i.e., reliability); anything you place at the lower levels (e.g., link) can be justified only on a performance basis. In many situations, attempts to make things more "reliable" at the link layer are redundant and just waste of bandwidth. Packet radio is about the only exception I can think of where link level reliability measures are justified on a performance basis. In virtually every other situation, links are either "up" (with a 10^-12 BER) or "down" (with .5 BER), so it isn't worth the effort.

If you really understood the Internet approach, you would realize that you can put almost ANYTHING (including VC networks) down below the Internet protocols. My horizons extend further than just AX.25 Level 2, and there are lots of other networks out there that I want to interoperate with transparently: local datagram networks (Ethernet, Pronet, packet radio) or long haul virtual circuit networks (ARPANET or X.25). It makes

no difference; a waxed string and a pair of tin cans are suitable if they can relay a datagram with some reasonable degree of reliability from point A to point B. These networks can and do coexist quite nicely in the Internet, because the Internet Protocol provides a uniform, standard service that hides the "seams" between these dissimilar networks for the higher level protocols while at the same time demanding as little as possible from the underlying network protocols.

So Howie, if you want to build a VC based network, fine! I certainly won't stop you, because IP can still use it. I'll nail up some virtual circuits through your network between my IP gateways and run IP datagrams over them. We do exactly the same thing with a Telenet X.25 connection right now, and that's how we access the rest of the ARPA Internet. Plus I can provide connectivity between your network and any other type of network people might want to experiment with. So it's really silly to argue; with IP we can all coexist and intercommunicate at the same time.

End-to-end acknowledgments are not provided at the Internet level; they are strictly the province of the transport protocol, which is a bilateral agreement involving only the endpoints of the "connection". Nobody requires you to use TCP if your application doesn't want or need end-to-end acks. You are free to develop your own transport protocol if you wish; the Internet couldn't care less. Link level acks aren't specified either; if the lower level protocol wants to provide them, that's transparent to IP also. Since the Internet layer is the only layer that everybody IS required to implement (in order to be interoperable), it was deliberately made as simple as absolutely possible -- hence the datagram-only service, since it's the least common denominator.

I don't understand your question about flow control. Conceptually TCP does per-VC flow control just like the packet layer of X.25 (which isn't a transport protocol, by the way, because it isn't truly end-to-end). Both use end-to-end (well, ALMOST end-to-end for X.25L3) windows and acks to control how much data can be in transit at any time. Other than where the protocols sit and how robust they are against network faults, they behave similarly.

From my own comparative experience, TCP, easily the most complicated single member of the Internet "suite", was considerably easier to implement than AX.25 Level 2. The IP layer/packet switch code is much simpler than X.75 (since there are no connections) and the only real complexity (needed in both our approaches) is in the automatic routing algorithms.

Regarding your question on congestion control: there are several aspects to the problem.

1. First, it is the end-to-end transport protocol that decides when to send a datagram, so it needs to be reasonably careful about doing this efficiently. For example, it is technically legal by the TCP spec to fill up a 2,048 byte receiver window with 2,048 1-byte TCP packets although it is clearly stupid to do so. A convention has been established in TCP implementations where only 1 packet will be allowed "in the pipe" at any one time, unless enough additional data can be sent to fill out a full-sized packet (in which case you don't save anything by waiting for more from the user). This technique works extremely well because it allows the transport protocol to put more user data in each packet without also slowing things down when the user needs lots of bandwidth.

2. Contrary to popular impression, the Internet Protocol is NOT a "pure datagram" protocol in the sense that a digipeater network is (i.e., send a datagram and pray, never getting any network-level indication when something is wrong). There is a mandatory adjunct to IP called ICMP (Internet Control Message Protocol) which reports errors to the sender of a datagram. If congestion occurs (or is about to occur) in an IP switch, it is required to send back one particular ICMP message called a Source Quench. This tells the transport protocol (e.g., TCP, but there could be others) to back off; the network is getting congested. Only if the user ignores Source Quench will the switches finally be forced to toss stuff on the floor. No ICMP message is sent unless there is a problem. Since most datagrams make it without a hitch this avoids a LOT of redundant overhead packets that just keep reassuring you that all is well with the network.

I think you'll find that virtual circuit network protocols are no panacea for congestion control, because congestion is always possible wherever you have shared resources. The only way to guarantee bandwidth during a connection is to abandon packets and use circuit switching; the only way to absolutely guarantee that you'll get a circuit when you need one is to put in a leased line. Whenever you share something, you can get congestion.

Yes, there are two fundamentally different approaches to building networks. But one approach (that of the Internet) is fundamentally superior if you're trying to support a loosely- (or un-)coordinated set of experimenters with a wide variety of top-level applications and lower level protocols but still wishing to intercommunicate.

You can't prove anything with metaphors. I admit I like them too, so I guess I'll add mine.

Unlayered virtual circuit nets are like railroads - track is expensive, but it provides a standard, high quality path. Virtual circuit transport protocols are therefore like railroad locomotives - good on fuel economy/ton but very poor on flexibility. They are highly specialized devices that only run on steel rails, and there are lots of places that steel rails don't go. VC nets and railroads work best when doing one very specific, customized application on a large scale where the lack of flexibility isn't a problem, like hauling coal from mines to powerplants or connecting remote dumb terminals to big IBM machines.

A datagram net is more like the road system. Some are Interstate highways, some are dirt roads, some are in between. They're owned and maintained by thousands of different organizations and private individuals. All are interconnected so I can almost always get from here to there. A transport protocol designed to operate on a datagram net is like an all-terrain vehicle - a general purpose device that can drive over almost anything. And you can still put it on a flat car and ship it by rail. (At the risk of really stretching the metaphor, my vehicle also has wheels so it can drive on rails, but the railroad forces me to rent a flatcar. Just like the X.25 carriers force me to buy extra from them, i.e., a virtual circuit, when all I want is a raw datagram service.)

Granted, the fuel economy of an all-terrain vehicle might not be as good as a locomotive per ton hauled. But let me ask you this: which would you park in your driveway as your SOLE means of transport?

Phil

C2974 CC110 Howard Goldstein (N2WX,2987) 12/11/85 9:49 PM L:39
KEYS:/QUENCH/SOUNDS LIKE A NEW SOFT DRINK/
A: 109

Phil,

I'm more concerned about the people reading this conference who didn't expect to see a 216 line reply! In the interest of brevity - seemingly an impossible goal in these discussions - can we confine this discussion to NETWORK and related issues? Don't care if it's interface or internet. Moreover, the sooner matters of semantics are dispensed with, the more meaningful dialogue can result.

With this in mind...

No one argues the utility of a broacast UI, but (you must know this!) the new UI in ax2512v2 does that and SO MUCH MORE. Alas, what "more" is supposed to be isn't clearly defined. That lone paragraph taken in the context of the rest of the document is vague.

Run anything on top of a vc net? Sure! Gateways? Excellent! And thank you for yielding on the requirement that EVERYONE run TCP.

This ax2513 bashing continues. Flow control in l3 is not almost end to end, it is ALWAYS hop by hop. The window is NOT preserved end unlike TCP WHICH IS TOTALLY END TO END.

Too, messages may be nested within themselves, for example an l3 ack can also ack at l2 in the same instant. Wish Eric were here to respond to your comment. Were 54 individual - in time and space - messages exchanged end to end?

I promised not to discuss transport issues but I thought you had proposed to make it's embellishments -- like ICMP -- mandatory throughout the network; not only at the endpoints. Anyway it seems now inextricably (sp) tied up with functions which are really the providence of lower levels, as with the matter of congestion control 'cause what happens when the quench - or other 'transport' units get lost?

Once again friend, it's End to End.

Non-sequitor? Hardly. Doesn't it depend on what "flavor" TCP you're running?

Did I hear somewhere that someone was running TCP over the phone lines? Guess a modem could be considered part of the link-layer. Possibly some of that group belongs to the other one that - conceptually, now - considers digipeaters the domain of the link layer. A NULL link layer is the same as one that isn't there - this has the same affect as doing away with same, and that's what was inferred.

Gosh this is long! Will pass on the mandatory methaphor despite the enjoyment everyone gets out of them.

Howie

C2974 CC111 Phil R. Karn (ka9q,2979) 12/12/85 12:19 AM L:93
KEYS:/AX25/SOUNDS LIKE A NUMBERING SCHEME FOR SLASHER MOVIES/

When people tell me that my comments are too long, I'm reminded of two famous quotes. The first is by Abraham Lincoln, who when asked how long someone's legs should be, replied "just long enough to reach the floor." The second is Albert Einstein's famous saying that "things should be simplified as much as possible -- but no simpler."

The issues of Network (3) and Transport (4) are so closely intertwined that it's impossible to discuss them separately. Indeed, it is the interface between these two layers (and which functions ought to go in each) that is so controversial. The basic position to which I subscribe is that certain functions (such as reliable end-to-end delivery and flow control) can ONLY be done properly at the transport layer. Therefore, attempts to do them at lower layers are at best duplicated efforts and at worst add enormous unnecessary complexity and performance bottlenecks.

I feel that we have made a major mistake in amateur packet radio by working upwards from the link layer; if we had defined a transport protocol FIRST and worked down, we'd have encountered this fact of life a lot sooner. Instead we're using AX.25 Layer "2" as though it were a transport protocol, and when we do finally come out with a "real" transport protocol all the users will have to change their habits (or go through Magic Protocol Conversion Gateways). KA6M first proposed this alternate strategy over FOUR years ago, and I really regret that Hank doesn't have a louder voice. I quote from his letter of June 10, 1981 to Paul Rinaldo, W4RI:

"...it [the Internet protocols] would save us an enormous amount of development time in bringing up our network. I'm really afraid that if we started with something much simpler, as some hams will encourage us to do, we would find in a year or two that features are lacking and extensions are necessary. Much discussion and haggling would produce a new and revised spec, and piece by piece we would eventually reinvent most of the features that are currently in the Internet document..."

As I said, the use of TCP as a transport protocol is a convention observed by the end hosts; the network doesn't care. However, it is THE protocol used when an end-to-end virtual circuit service is needed, so you are not likely to be able to talk to many services unless you have it. That's why I'm concentrating my time on an implementation which I am giving away to the amateur community.

I probably don't understand your implementation. I was assuming earlier that you implemented your level 3 in the manner described to us a year ago by Eric Scace, in which layer 3 flow control packets are passed unchanged (except for the translation of VC numbers) across X.75 gateways from end DTE to end DTE. He explained that this is how Telenet operates internally. Layer 3 flow control is therefore "almost" end-to-end, or, in other words, the network behaves as though the "D" bit is always on.

While I've cursed at them for this, it is only because of the tiny window values they chose. If you are instead relying solely on link-level "backpressure" then I think you're going to be in serious trouble. As soon as people start shoving big files into the network without a transport protocol, they'll likely "gridlock" every TNC in the path. So they'll still need some sort of end-to-end transport protocol that limits the number of outstanding packets in the network, and you may also require a "source-quench" mechanism in case too much traffic from different sources happens to pass through a bottleneck.

I was just counting the sendmsg/recvmsg pairs in the Q.931 SDL diagrams. If you are on the Digital Committee mailing list, you will most likely receive a copy. These represented the transactions between originator and transit switch, and transit switch and destination. They were not all end-to-end.

ICMP is implemented as a "pseudo-transport" protocol in that it has an IP protocol type assignment as does TCP, but it is really better thought of as a part of IP. ICMP messages are generated by gateways (IP packet switches), which do not have to implement any other protocols like TCP (unless they are also hosts, of course). ICMP messages are purely ADVISORY. A transport protocol running atop IP has no guarantees that all its datagrams will be delivered, nor is it guaranteed an ICMP message should a datagram be dropped (ICMP messages are themselves datagrams, and datagram delivery is never guaranteed). Therefore, a transport protocol such as TCP must still use positive acknowledgments with retransmission to assure reliability. ICMP messages only help to speed things up, or to help diagnose a network problem. For example, if TCP receives an ICMP "destination unreachable" when trying to establish a connection, it doesn't have to retry N times before giving up; it can notify the user right away. If the network becomes congested, TCP will likely encounter higher packet loss rates and begin backing off its retransmissions anyway; an ICMP source quench just starts the process early, before the network is forced to resort to dropping packets.

I don't understand your last paragraph, the one about modems being part of the link layer. Actually, I think you'll find that although real networks are layered, the pieces seldom conform in a nice one-to-one fashion to each of the 7 ISO layers. The OSI model is supposed to be a rough, descriptive GUIDE for understanding existing networks, not a mandatory straightjacket for implementers. There is a cute "proof" in one of the USC-ISI papers that, according to the ISO model, N = N + 1 for any value of N. The point they were making is that layers are relative. Even the Internet protocols don't go far enough, in that I believe real networks are becoming multi-level hierarchies of sub- and super-networks, and the original Internet model only allows two (subnet and internet).

Phil

C2974 CC112 Howard Goldstein (N2WX,2987) 12/12/85 10:39 PM L:19
KEYS:/AND VC'S ARE A LOT OF FUN TOO/
A: 111

Phil,

My implementation supports the "D" bit OFF only - not a major decision really because `almost end-end' acknowledgements, D=true, is specified as an OPTION to be negotiated at call setup time (in X.213 network service specs) and define hop-hop as default.

Similarly, "backpressure" is applied on a HOP BY HOP basis, at or before the time translation takes place to the other LCN. Since the links themselves never get "stopped up", they don't gridlock.

TELENET's decisions don't have to make technical sense. How they implement has as much to do with ham packet as it does with concern for elegance and flexibility. On the other hand, we're Amateurs having one hell of a great time and are unconcerned with overhead and capital cost.

I'm not sure how my example of a modem plus "TCP and the sound of electrons" as kind of a null level 2 was understood as a mandate to try to implement the model in contradiction to its own directive NOT to understand it as a mandate....

C2974 CC113 Phil R. Karn (ka9q,2979) 12/12/85 11:27 PM L:33
KEYS:/BUFFERING AND GRIDLOCK/

Howie,
I believe the reason Telenet did it the way they did is to guard themselves against congestion by limiting the number of outstanding packets on any single virtual circuit. Since the predominant use of their network is for slow speed remote terminal access, the throughput limit this creates is not a real problem. It *IS* a major hassle when doing host-to-host file transfers, though, which is why we go to such lengths to circumvent it. Of course, our technique for circumventing it runs directly counter to their reason for doing it in the first place, and since they don't have a "source quench" mechanism to tell when their network REALLY gets into trouble, there's nothing we can do to help. I wouldn't mind helping out if they didn't ask us to cut back unless it was really necessary, but I do object to having lousy throughput all the time. It's like not being allowed to carry on more than two pieces of luggage on board a plane, even when it's half empty and there's plenty of space.

How do you do link level flow control when handling multiple simultaneous links? Unless you do it by static allocation of X amount of buffer space to each link connection (whether or not it is used) you might encounter some rather nasty gridlocks, e.g., where A can't send to B because B's buffers are filled with traffic to C, who can't accept them because its buffer's are filled with traffic to A, and so on. If you do dynamic buffer allocation (necessary to handle a large number of links on a small machine) and don't take extra precautions, especially when traffic might sit in a given machine indefinitely, then I think you're leaving yourself open to deadlock.

I see that you have two choices: either use dedicated buffers (and deny additional links, even if the existing ones are idle) or allow many more links with dynamic buffer sharing, and take the risk of having to drop a packet occasionally when things get busy.

Phil

C2974 CC114 Phil R. Karn (ka9q,2979) 12/13/85 1:38 AM L:7
KEYS:/CONGESTION CONTROL/

Howie, if you have a copy of Tanenbaum ("Computer Networks") take a look at the section on Congestion beginning at page 215. He provides a pretty complete description of the buffer allocation and deadlock problems, along with their most common solutions. Using his terminology, the ICMP Source Quench is a "choke packet".

Phil

C2974 CC115 Howard Goldstein (N2WX,2987) 12/13/85 8:05 AM L:23
KEYS:/BUFFERING AND GRIDLOCK/HOW MUCH OF AN IMPROVEMENT IS IMPLEMENTATION DEPENDANT/
A: 113

Phil,

Granted that if it was done, flow control solely through the link level would cause the gridlock problem you describe.

But flow controlling at that level is NOT the desired end, I don't think. At least that's not what I'm shooting for.

RNRing the link is prevented directly at the link level by insuring that enough buffer space exists to process a level 3 packet. If packets are processed individually, which they are, this buffer need only be the size of the largest packet one will ever expect. Link flow is no longer an issue.

Now let's get down to some (implementation specific) nitty gritty. Static allocations of X amount of buffer MUST take place at call setup time and for each hop in the network - but X can be a very small number, maybe 20 or 30 bytes, since we're only concerned with preserving the state of the level 3 interface, and are never forced to accept data units.

How a participant in the network arranges to allocate for network data units is IMPLEMENTATION DEPENDANT and subject to local change through improved hardware or whatever.

I'm not claiming VCs are any panacea, only that through their use we can build a highly usable network with the restrictive development and destination tools available today (eg. 820) yet still be prepared to offer even more in the way of throughput and services by way of the better processors and/or greater memory space we'll have tomorrow.

C2974 CC116 Phil R. Karn (ka9q,2979) 12/13/85 8:42 PM L:29
KEYS:/BUFFER MANAGEMENT/

Howie, maybe it would help if you described your implementation in more detail, so I can see how you solve the problem. It's NOT just an "Implementation Dependent Detail", because certain approaches require features to be in the protocol that might not be there.

You're right, static allocations can only be done with virtual circuit protocols, but that's not the only way to do it. Static allocation has the advantage (if done right) of completely eliminating all possibility of deadlock, but it does so by turning a "packet" network (where things are supposed to be allocated on demand) onto a "pseudo-circuit switched" network, where resources are busy whether or not real traffic is flowing. To me, this isn't true "packet switching". If you want to build a circuit-switched network, fine; they have plenty of appropriate applications, but bursty, interactive data networking isn't one of them.

The only way you can guarantee your static allocations is by refusing additional connections when you run out of memory, even if your switch hasn't seen a packet for three days. You may want to consider this point, for no other reason than that I will be coming along and nailing up virtual circuits over which to pass datagrams, and I don't want to have to bother with timing out unused circuits and re-establishing them on demand.

One of the things I like being able to do is to stay logged into a remote machine all weekend without having to log out, because I consume absolutely no network resources by doing so. This is not the case with VC subnets, at least not without considerable complications with session-layer "reconnects" and other similiar stuff.

C2974 CC117 Howard Goldstein (N2WX,2987) 12/14/85 9:33 AM L:20
KEYS:/AND NOW FOR SOMETHING COMPLETELY DIFFERENT/REAL SYSTEMS/
A: 116

Phil, your statement in re: implementation dependent detail is non-sequiter. Recall there is a minimum, mandated functionality which is set forth in the osi network service definition.

Are you not stretching it a little bit on allocations? When an IP net drops packets, it might or might not get a message back to the endpoint through an indeterminate # of nodes saying "shut up." And then more and more resending until the throughput approaches the zero point.

It makes a lot more sense to shut up a neighbor than to shut up an endpoint thousands of miles away.

I feel its worth 20-30 bytes per call to do it right (i.e., locally). Besides, memory is cheap (right?), plus you run into throughput constraints way before you'll run out of 20 to 30 byte blocks. I figure at that rate you run out of static allocations after something like 300 calls/interface.

If we've come to arguing over minor turds like this, then the dialogue has surely been worthwhile.

Oh, yes, if you are concerned about timouts then why don't you just treat the network as a big datagramme system and pump restricted response call requests through the damn thing? I see we can argue like this for many more months - perhaps years; But I want to get on with implementing real things.

C2974 CC118 Phil R. Karn (ka9q,2979) 12/14/85 7:39 PM L:52
KEYS:/CONGESTION CONTROL/

>Recall there is a minimum, mandated functionality which is set forth in the
>osi network service definition.

The ISO RM is supposed to be a GUIDE for describing existing networks, not a mandatory set of requirements for implementing them. Attempts to do otherwise belong in the same category as attempts to legislate the value of pi.

An IP gateway may choose to return an ICMP Source Quench when it sees trouble brewing, in advance of actually having to drop packets. If the host properly obeys the protocol, then there won't be "more and more resending until the througput approaches the zero point". It is true that in a datagram network an improperly functioning transport protocol module (one that ignores source quenches and/or the far end window) and floods the network can cause problems, unless a nearby gateway notices the problem and takes appropriate action (like dropping all its packets). However, similar things can happen in a VC network. What do you do with a station that decides to flood the network with call setup requests? Congestion control is a much more fundamental problem than the choice of network protocol.

>It makes a lot more sense to shut up a neighbor than to shut up an endpoint
>thousands of miles away.

It makes more sense to me to have an end-to-end protocol that keeps packets out of the network unless it seems likely that they'll be delivered quickly to (AND accepted by) their destination. Just like a highway, a packet network's primary job is transportation, not storage. You're perfectly free to use link-level flow control on a datagram network. We do that but have found that it doesn't buy much. It usually just gets in the way by generating twice as many interrupts per second that the CPU has to process.

I'm confused by just how much you allocate per VC. Obviously you need a control block (20-30 bytes) but aren't you also pre-allocating your DATA buffers? If instead you work out of a shared buffer pool, you MUST have rules that intentionally drop packets whenever necessary to ensure having buffer space for incoming link level acks that would free up other packet buffers. Otherwise you leave yourself open to deadlock. Consider the following scenario, known as "store-and-forward deadlock":

Station A has N packets for station B, but has no buffers left to accept an acknowledgment from station B. Station B has N packets for station A, but has no buffers left to accept an ack from station A.

Plenty of variations exist; you could have a ring of stations instead of just two all waiting for events that never happen.

I *could* use your network in "fast select with restricted response" mode, but I suspect it'll perform very poorly. I'd find it easier just to nail up a circuit so I don't have go invoke all your setup overhead on each and every datagram packet.

Phil

M 23591 Lyle Johnson (WA7GXD,2973) 12/16/85 11:16 PM L:19
KEYS:/TNC2 REV2 NOTICE/PLEASE POST ON PBBS'S/
TO: WA7GXD, (Group 95)

To: All TNC 2 Rev 2 Owners
From: TAPR
Re: Minor Hardware Bug

The Rev 2 PC board contains a new resistor, R98. This resistor provides a software-readbale input directly from the modem for future use with AMTOR, RTTY and the like. It is supplied as 4.7k ohms.

Unfortunately, the SIO chip used at U22 provides a pulse output on the same pin that R98 is attached to when the SIO is running in HDLC mode (as it does on packet).

The result is possible interference with State Machine operation.

Please change the value of R98 to 100k ohms for proper packet operation.

Thank you.

C2974 CC119 Howard Goldstein (N2WX,2987) 12/16/85 12:18 PM L:23
KEYS:/CONGESTION/SEMANTICS/X.213/
A: 118

A TCP/IP free for all network flooded the way you describe it would approach zero throughput a lot faster than a VC type. As congestion builds, or with low reliability links slowing everything up, end-end timers of TCP are invoked and more flooding ensues. Chaos prevails until connections are forced down.

A VC switch can simply reject new calls it can't handle if it's being flooded, passing the onus to the preceeding switch or the calling transport layer for redirection.

Won't an approach where the responsibility for congestion control at an interface is done locally, by the interfaces in question, work best simply because the end users of the network don't have to get involved in remote clogging issues?

I don't understand how the truism that ISO RM is a model has anything to do with the (out of context!) quote taken:

>"Recall there is a minimum, mandated functionality which is set forth in the osi network service definition."

This most clearly refers to X.213, a "NETWORK SERVICE DEFINITION FOR OPEN OPEN SYSTEMS INTERCONNECTION FOR CCITT APPLICATIONS" - a SPECIFICATION, and not a ```guide for describing existing networks''' (although I wouldn't be surpirsed if it had a part in aiding some design open systems).

Whether it falls into the category of ```attempts to legislate the value of pi''' is as much an argument against a TCP/IP/ICMP spec as it is AX.25 or any other...

C2974 CC120 Phil R. Karn (ka9q,2979) 12/16/85 1:14 PM L:50
KEYS:/CONGESTION/ISORM/

I'd like to send you a copy of ARPA RFC-896, "Congestion Control in IP/TCP Internetworks," by John Nagle of Ford Aerospace. He has come up with some ideas that are very good, and have virtually eliminated the phenomenon you describe. My TCP uses his algorithm (which limits the number of outstanding packets on a connection to 1 unless there is enough data to generate more maximum-sized packets), plus it has rround-trip timing with an exponential backoff strategy so that it 1) adapts its retransmission timer to the actual time needed to acknowledge a segment and 2) backs off very rapidly on successive retransmissions when things don't seem to be getting through. These algorithms combined do very nicely when working between two Ethernets connected by a slow (1200-9600 baud) link.

The problem with your technique for congestion control is that it rejects calls even when the packet switch involved hasn't seen a data packet in three days because it is filled with idle virtual circuits. Besides, even limiting the number of virtual circuits won't prevent deadlock unless you've made static buffer allocations to each VC, and at the rate of a few K per VC you run out pretty quickly.

I don't care what illusions of grandeur the ISORM-ites (term attributable to M. A. Padlipsky) have, their model is only that -- a model. Consider the following quote from "The ISO Reference Model and Other Protocol Architectures" by Postel and Cohen:

"We accept the ISORM as a great contribution for teaching about the underlying concepts of computer communication. On the other hand, we cannot accept it as a model for all seasons, as an implementation specification to be followed to the letter in all situations.

"Note the striking similarity between these statements:

* In 1978, ISO invented the ISORM; this made computer communication much easier.

* In 1777, Lavoisier invented oxygen; this made breathing much easier.

"We do not attribute such statements to the OSI committee, but to camp followers who have misunderstood the ISORM and its significance. Just as Lavoisier's invention of oxygen did not change breathing, the ISORM did not change the approach to communication. At best, it helped some people understand it better.

"According to Babor and Lehrman, oxygen was first discovered by Joseph Priestly, who isolated it from mercuric oxide and reported it on August 1, 1774. In October 1774, Antoine Laurent Lavoisier started repeating Priestly's experiments. After three years of work, Lavoisier published his report on oxygen, for which he claimed eternal credit. As noted above, the similarity to the ISORM situation is striking."

Phil

C2974 CC121 Howard Goldstein (N2WX,2987) 12/17/85 7:39 AM L:6
KEYS:/PADLIPSKY POSTEL AND COHEN/ABRAHAM MARTIN AND JOHN NEXT?/
A: 120

Padlipsky, Postel, Cohen...

Have these guys done anything for amateur networking?

The analogy was enjoyable, but - as the Georgia SOUTHNET folks may be fond of saying - you can't put all those papers, even combined, on a mountaintop and make them work.

C2974 CC122 Phil R. Karn (ka9q,2979) 12/17/85 1:49 PM L:28
KEYS:/WHAT HAVE THEY DONE FOR US?/

Well, let's see...
From Gateway, Vol. 1, No.4, Sept 25, 1984 (that's over a year ago):

"We received the following letter from Richard Bisbey II, NG6Q:

"'The Information Sciences Institute Amateur Radio Group (WB6MXZ) has been on the air with a 56 Kb packet radio system since December, 1983. The equipment consists of 8088-based controllers using the Zilog 8530 SCC and frequency agile 10-watt FSK transmitters and receivers. The controller supports serial, parallel, and Ethernet connected devices. Assembled controllers cost approximately $650. The system uses standard INTERNET (TCP/IP) protocol, and in addition to supporting simple link level connections, supports internetworking as well as numerous high-level protocols such as TELNET, File Transfer (FTP), Mail (SMTP), Multi-Media Mail (MMM,MPM), Graphics (GP), Packet Voice (NVP) and Remote Virtual Disk (RVD). The hardware and software can support data rates as high as 1 Mb well in excess of the current amateur limit of 56 Kb..."

They go on to talk about a transcontinental internet test from an airplane, and a demo of intercontinental "internetting" from a packet radio in California to Europe via satellite. They're also working on a low-cost spread spectrum version of the system.

We don't hear much from these guys mainly because they see the rest of us as hopelessly stuck back in the Stone (well, maybe Bronze) age, and they're absolutely right.

Phil

C2974 CC123 Lyle Johnson (WA7GXD,2973) 12/17/85 9:19 PM L:8
KEYS:/NNC PROGRESS REPORT/

All,

Tonight the prototype NNC successfully interfaced to a Qume 142 Floppy Disk Drive and booted (and ran) Z-DOS under Z-CPR3.

I know this sounds pretty trivial, but I am excited!@

Lyle

C2974 CC124 Skip Hansen (WB6YMH,2964) 12/17/85 10:48 PM L:24
KEYS:/PR-BS1/

Phil,
I usually stay away from the Datagram vrs VC wars, but your last comments on that old Gateway article hit a sour note!

1. The airborn "test" was on a unmodified VADCG board using unmodified VADCG software which was for a large part made operational by WA6JPR who received no credit in the various "PR" releases. The other side of the historical link was simply pluged into a modem which happened to be dialed into ARPA. BIG DEAL!

2. The 56K baud modem was shown at a ARRL convention in 1984, but none of the information was ever made available or published in any amateur publications even after numerious requests.

3. A large part of this work was done under research funding.

4. The work these guys do is great and does make us look like we are in the Stone age, but remember these are paid professionals who also happen to be Hams and are lucky enough to get paid to do what we do for fun.

Yea, I wish things were developing faster too, but unfortunately I work for a living where packet radio is not a company project. 73's Skip

C2974 CC125 Phil R. Karn (ka9q,2979) 12/18/85 1:38 AM L:48
KEYS:/SOMETIMES WHEN YOU REINVENT THE WHEEL YOU GET BACK A FLAT TIRE/

Skip,

I'm sorry I hit a sour note, I knew very little more than what I read, plus a few mail conversations with the people involved. But to answer a few of your comments:

1. It would seem that the ability to run TCP/IP over even an unmodified VADCG board and to access all those services on the ARPANET without the other ends being aware that a new form of packet radio link was involved says a lot for the design of the ARPA protocols. I have yet to see an alternative approach that can claim the same flexibility.

If you can handle a variety of new jobs effectively with the tools you already have, I think I'd respect your toolmaker more than if his products were so specialized that you have to buy a whole new set for each new occasion. Unfortunately, toolmakers only make money by selling NEW tools, so...

2. If what you say about the modem is true, that's a real shame. We desperately need such a critter.

3. So what? Lots of things hams now have were first created under research funding by corporations or the government. Just because Zilog spent millions developing the Z-80 doesn't mean we can't benefit from it; they spent the money so we don't have to spend it ourselves. I guess it's the old "I built it all myself out of junkbox parts" syndrome, but of course the "REAL" hams build all their own capacitors and rectifiers too. Personally, I don't get much out of wheel (or protocol) reinvention. I prefer to spend my time building things I CAN'T buy off the shelf.

Sometimes just knowing that somebody has been able to do something is in itself very valuable. Even if you don't get the details, at least you know it's POSSIBLE. I know from my own work that much of a researcher's time is spent persuing dead ends. You're there because nobody else has been there before, and you never know if something will pan out until you try. If on the other hand you're trying to develop (or duplicate) an operational system out of existing technology, then doesn't it make sense to listen to people who have "been there" so you don't repeat their mistakes? You are certainly free to view their advice with a jaundiced eye and not follow it, but you shouldn't reject it out of hand just because of who funded it.

I keep sensing the widespread attitude that amateur packet radio is somehow totally unique in the world, and nothing that anyone did on the outside (most of all with government money) could possibly be relevant to our needs. This seems to be an awfully wasteful approach, especially when as you say, we aren't lucky enough to earn our livings off it.

Phil

C2974 CC126 Thomas A. Moulton (W2VY,995) 12/18/85 2:19 PM L:10
KEYS:/CREDIT WHERE CREDIT IS DUE/
A: 125

A:120
Howie, FYI RFC stands for Request For Comment, which means it's a Standard?

Phil,
In your last comment you praise ARPA for being able to plug in a VADCG board into an ARPA plug instead of the regular modem, I think you praise was sent the wrong way, after all these computers are connected via RS-232 interfaces and a packet board looks just like a TEL CO modem!

The basic function of the current packet boards is a point to point link

C2974 CC127 Phil R. Karn (ka9q,2979) 12/18/85 9:30 PM L:70
KEYS:/PROTOCOLS/PROGRESS REPORT/

Howie, there is a timely and relevant paper in the December 1985 issue of IEEE Transactions on Communications: "Prevention of Store-and-Forward Deadlock in Computer Networks," by Inder S. Gopal. You might be interested. I can forward a paper copy if necessary.

Tom,
Even point-to-point modems vary widely in link speed, error rate and control interfaces. If the higher level protocols provide a very simple interface to the lower levels (e.g., link) and assume the minimum possible grade of service, you can use a waxed string and a pair of tin cans if it's capable of getting packets from here to there with > 0% probability.

All:
I am continuing work on my TCP/IP package. At this point, the pieces are

1. Low-level buffer primitives, utility functions, etc. (1.5 K on 8088) Essentially stable.

2. Transmission Control Protocol (TCP) module. (5.5K on 8088). Recently added support for Internet precedences (Routine, Priority, etc). Tested and working, no significant changes planned.

3. Internet Control Message Protocol (ICMP) module (1K on 8088). Recently written, under test. Common options are supported; some esoteric ones I haven't bothered with. Need to add proper Source Quench support.

4. User Datagram Protocol (UDP) module (750 bytes on 8088). Written, but untested. Needs redoing to provide more information to the user.

5. Internet Protocol (IP) module (1K on 8088). Supports sending and receiving IP datagrams on behalf of transport layer protocols. Recently added static table-based routing with primitives for adding and dropping entries from the routing table; makes source routing (complete path specification by datagram sender) unnecessary. No fully-automatic routing protocol is in yet; this will take more thought and work than I have at the moment. Haven't implemented full IP-level fragmentation/reassembly or IP option processing yet.

6. Serial Line IP (SLIP) module (400 bytes on 8088). Provides a convenient way to transfer IP datagrams across conventional asynchronous RS-232 interfaces, either hardwired lines or dialup modems. Tested and stable.

7. AX.25 link layer module (not rewritten yet). I am planning on writing a new AX.25 link layer for IP use, since I have come to the conclusion that the complexity of the complete LAPB protocol is unnecessary when just passing datagrams from point A to B. Each datagram will be sent in an AX.25 UI frame, and if link level acks are desired, the Poll bit will be set to request a one-for-one UA ack from the other end. To avoid the introduction of duplicates, I am planning on having a simple fixed-length FIFO list of datagram headers recently received by each IP module so that duplicates can be recognized and discarded without actually having to maintain lots of link level connection state.
This stripped-down link layer will also have to support the Address Resolution Protocol (ARP) for mapping IP addresses into AX.25 addresses without the need for a manually maintained lookup table.

Well, that's it from here. If anyone would like to play with this code, you're welcome to it, but unless you're serious about actually hacking on it I'd recommend that you wait until it's more complete so I don't have to send out too many updates. What's been written so far has compiled and run in "self test" mode on about 4 different processor types without portability problems, although I'd like to find people willing to flush out any remaining portability problems with other C compilers.

73, Phil

C2974 CC128 Howard Goldstein (N2WX,2987) 12/18/85 9:33 PM L:2
KEYS:/UI/AND WHAT IS "SELF TEST" MODE?/
A: 127

No wonder

C2974 CC129 Howard Goldstein (N2WX,2987) 12/18/85 9:44 PM L:3
KEYS:/RFC/MAIL/
A: 126

Thanks Tom, I guess I should know that but you know how tough it is to remain well connected when all your mail gets lost (like they were datagrammes)

C2974 CC130 Phil R. Karn (ka9q,2979) 12/19/85 12:58 AM L:12
KEYS:/SELF TEST/ASK A VET IF HE LIKES VC'S/

Self-test mode (dubbed "incest-is-best mode" by one of my cohorts) is where you loop back the IP datagrams in software, create two TCP descriptors, establish a connection between the two and send some test data back and forth. It exercises just about everything in the TCP logic including the provisions for simultaneous connection establishment and closing, which almost never happen in real life.

I'd like to ask how many banks out there trust an X.25 network's assurances of reliability enough to do away completely with all forms of end-to-end checking on their $500,000,000 funds transfers...

Phil

C2974 CC131 Skip Hansen (WB6YMH,2964) 12/19/85 1:11 AM L:34
KEYS:/MORE PR BS/

Phil, I guess I shouldn't have started out my previous message by referencing the Datagram vrs VC wars. I only wanted to point out that the cited reference was in my opinion "PR hype" and that the actual experiment was not quite what one would assume from reading the Gateway (and other) writeups. I object to the comparision of the accomplishments (real or otherwise) of the Ham groups being compared with that of professional groups. I have nothing against them because of their TCP/IP leaning. The whole "test" just reminds me of when we got our first autopatchs going, we would impress the unknowing by calling information in the deep south (interesting sounding accents to Southern Californians) to ask what the weather was. The "acomplishment" was in getting a dial tone and being able to dialout, not in the distance covered by Bell after we dialed!

If the ISI guys had plugged their TNC into a different modem jack a X.25 circuit connection to Europe would have been just as impressive (or unimpressive!). TCP/IP was not used over the TNC link at that time so protocol flexibility was not tested.

As for not re-inventing the wheel, I am all for it when the previous wheel builders either publish the information on the design of the wheel or leave it around somewhere where it could "reverse engineered" It is hard to avoid anothers researchers mistakes when they are never published.

I am also specifically not saying that ISI hasn't developed some very worthwhile stuff after all, as I am sure you know, some of the ISI players were the "inventors" of ARPA net (their words not mine).

If "real hams" build their own capacitors maybe I am mistaken about ISI, they grow their own silicon!

And by the way of course I am a Trekie and proud of it. No offense taken. 73's Skip

C2974 CC132 Phil R. Karn (ka9q,2979) 12/19/85 3:17 AM L:24
(ORIG.) 12/19/85 3:13 AM L:22
KEYS:/AMATEUR PSYCHOLOGY/FOOD FOR THOUGHT/

Here's another thing to keep in mind while arguing the merits of different ways to build amateur packet radio networks.

The average amateur is very individualistic. It is much more difficult to get him to contribute to a joint effort resulting in a shared system than it is for him to buy another piece of hardware for his shack. AMSAT sees this all the time; the same person who gladly forks over $2000 for an HF KW linear he hardly needs balks at buying a $100 share of a communications satellite that will likely do far more for his communications capability. The linear he can park in his shack, twiddle the knobs and show off to his friends. The closest he ever gets to the satellite is a picture in the Amsat Satellite Journal; it isn't truly "real" to him.

What this says to me is that a network design philosophy that puts the lion's share of the "smarts" out in the "network" (as opposed to the amateur's shack) is going to rub against the grain of most amateurs; they won't support it. Oh, they'll all say they want it alright, until you ask them to kick in extra money to help support it. Given how much money most hams have invested in RF hardware just to avoid paying the telephone company, I suspect you'll find it easier to sell him a more expensive box than it'll be to sell him a cheap box that requires a bigger membership fee to talk beyond his own back yard.

Phil

C2974 CC133 Howard Goldstein (N2WX,2987) 12/19/85 8:15 AM L:12
KEYS:/QOS/IF IT COST $50 FOR AN OSCAR.../
A: 132

I don't see your point PHil, encryption and end-end reliability are, clearly, transport issues. The `flavor' of network running underneath is not germane (although the QOS might be).

I feel the argument that a VC network is too expensive is unfounded. There are hundreds of $50 Xerox 820s out there in (how shall I say this?) 'communal' amateur service.

Again, this point is rather unclear Phil since on the one hand you've claimed VC switches are more $$ to implement while on the other hand, you deplore the fact that IP embellishments, on-the-fly routing of each and every packet etc. are missing.

The very things you're touting would force the switch's cost up to beaucoup (sp) bucks!

C2974 CC134 Phil R. Karn (ka9q,2979) 12/19/85 12:32 PM L:7
KEYS:/HAM PSYCHOLOGY/

Note that I didn't mention "datagrams" or "virtual circuits" in that comment. I was only saying that if you have a choice between putting a certain function in the network or in the user's host, one would do well to consider the psychology of the average ham. Such functions might include reliability measures, protocol translations, etc.

Phil

C2974 CC135 Thomas A. Moulton (W2VY,995) 12/19/85 3:46 PM L:22
A: 134

I think that psychology is fading, after all see how many people want to have 2M repeaters up?

In any case I think that technically the more dispersed the network functions are the better off we will be, How many hams want to buy an MS-DOS system (aka 8088) for their Packet station?
Hell we have a hard enough time with getting them to buy anything other than a C64 or other Home computer (as to a Personal Computer)

And look at the people who got into packet with the software approach!

I think that we should remember that hams are cheap, lets get them in for a smaller box in their station.

after all don't the larger percentage of the USERS like to be part of the Club? (as in local club)

So!
I guess this still means,
You go your way, I'll go mine, I'm sure our packets will meet!

Happy Holidays ALL!

C2974 CC136 Phil R. Karn (ka9q,2979) 12/19/85 8:18 PM L:3
KEYS:/THOUGHT FOR THE DAY/

Those who believe that robust transport protocols are unnecessary because the network should be made "reliable" probably also believe that Star Wars will work.

M 24720 Gwyn Reedy (W1Bel,2975) 12/20/85 6:09 AM L:10
KEYS:/REQUEST FOR ARTICLES/
TO: (Group 95), W1Bel

ANNOUNCEMENT

The PSR Quarterly for January is undergoing construction at this time. There have been major inputs by WA7GXD and very little from anyone else. Please consider doing an article for the PSRQ during the beginning of the holiday period and get it to me by Dec 27th. Also I request you let me know that you are planning an input so that I may reserve some space for your work. Thanks.
Gwyn

C2974 CC137 Thomas A. Moulton (W2VY,995) 12/20/85 9:30 AM L:7
KEYS:/ROBUST/
A: 136

Phil,
I think we do BOTH have Robust protocols,

Robust: 1) strong and healthy, 4) full and rich
2) suited to or requiring physical strength or stamina
3) rough; coarse; boisterous

C2974 CC138 Phil R. Karn (ka9q,2979) 12/20/85 1:35 PM L:6
KEYS:/ROBUST/

I thought that in the context of a communication protocol, "robust" means that it's able to handle successfully a wide variety of unpredictable network behavior, including loss, misrouting, duplication, (often with large delays between the original copy and the duplicate) and misordering, without data being lost or erroneously delivered to the user.

C2974 CC139 Thomas A. Moulton (W2VY,995) 12/20/85 9:43 PM L:6
KEYS:/ONLY TIME WILL TELL/YOU ARE ABOUT TO SEE AN X.400 EXPLOSION/
A: 138

Again, both are robust, TCP/IP and X.400/X.200
They will both work.
Each has it's technical advantages.
Each has it's draw backs

ar sk

C2974 CC140 Phil R. Karn (ka9q,2979) 12/21/85 2:47 AM L:12
KEYS:/THERE'LL BE AN EXPLOSION ALRIGHT/

I assume the explosion will occur when the users find out just what a bill of goods they've been sold. A "standard" that consists of five co-equal but mutually incompatible "classes" is no standard. Everybody but the European PTTs are already talking about punting everything but the TCP-lookalike TP-4. But if you're going to do that, why bother changing in the first place (unless the vendors see it as a chance to sell you a complete new set of tools...?)

PACIFIC PACKET RADIO SOCIETY
Post Office Box 51562
Palo Alto, CA 94303

SAN JOSE
-PM
NOV 30
1986

H.S. Magnuski 86
2019 Barbara Drive
Palo Alto, CA 94303